Trabajo de Fin de Grado
Ingeniería de Telecomunicación

Inyección de malware en aplicación Android legítima

Autor: Manuel Jesús Gutiérrez Fernández

Tutor: José Manuel Fornés Rumbao

Dpto. de Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2019

Proyecto Fin de Carrera
Ingeniería de Telecomunicación

Inyección de malware en aplicación Android legítima

Autor:
Manuel Jesús Gutiérrez Fernández

Tutor:
José Manuel Fornés Rumbao
Profesor titular

Dpto. de Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2019

Trabajo Fin de Grado: Análisis de herramientas de inyección de malware en dispositivos móviles con Android

Autor: Manuel Jesús Gutiérrez Fernández

Tutor: José Manuel Fornés Rumbao

El tribunal nombrado para juzgar el trabajo final de grado arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2019

Abuela, no lo has visto,
pero lo conseguí.

Resumen

Los dispositivos móviles que llevamos en nuestros bolsillos son una fuente de información enorme. Tenemos desde fotografías de todo tipo hasta información muy importante como cuentas bancarias. Es por eso por lo que los dispositivos móviles se han vuelto un objetivo para los crackers o hackers con malas intenciones.

En este proyecto se muestra lo fácil que puede ser crear tu propia aplicación infectada para dispositivos Android y conseguir instalarla en otro dispositivo con la ayuda de una serie de herramientas e ingeniería social. Se explicará el proceso que se ha realizado para infectar un dispositivo y robar su información, así como todas las herramientas que se han utilizado.

Abstract

The devices we bring in our pockets are a huge information source. We have from pictures to very important information like bank accounts. The mobile devices are becoming very important to crackers or hackers with bad intentions.

In this project we are going to show how easy It's to create your own malicious application to Android devices and to install in another device with some tools and social engineering. We are going to explain the process we have done to infect a device and steal the information, and all the tools we used to get it.

Índice

1 INTRODUCCIÓN

La información que albergamos en nuestros dispositivos móviles es más valiosa que lo que la gran parte de la población puede imaginar. La información es poder y es por eso que los hackers con malas intenciones, los llamaremos crackers de aquí en adelante, buscan atacar a los lugares donde más información almacenamos. ¿Cuál es uno de esos lugares? Los dispositivos que siempre llevamos encima, nuestro smartphone.

La *Diagnostic and Statistical Manual of Mental Disorders (DSM)* ya ha reconocido la *nomo- fobia* (miedo a sentirse desconectado de Internet) como trastorno mental que va en aumento entre los poseedores de dispositivos móviles. Lo que llama más la atención de los crackers en busca de información o de amenazar al usuario con mostrar cierto contenido sensible que haya sustraído del dispositivo de la víctima.

Los crackers siempre están buscando maneras de robar información a los usuarios poco concienciados con la seguridad de su dispositivo. Es por esto por lo que no es difícil encontrarte con multitud de herramientas en internet que están abiertas a cualquier persona que las busque. Desde códigos fuente de troyanos que funcionan en las versiones más recientes de Android hasta herramientas todo-en-uno que directamente te inyectan el malware en la aplicación y lo conectará a la dirección IP que le indiques.

Todo sistema operativo está expuesto a ser atacado con el fin de vulnerar la seguridad que ofrece. Android es uno de los sistemas operativos más bastamente encontrado entre la población mundial. Ni que decir tiene que iOS, el sistema operativo de Apple también cuenta con vulnerabilidades que los crackers aprovechan para sustraer información.

Ser un sistema operativo que encontramos fácilmente entre los usuarios de smartphones hace que los crackers centren sus esfuerzos en encontrar una manera de sustraer información. En internet, foros y páginas especializadas, no es difícil encontrar un buen tutorial donde te expliquen cómo inyectar un virus en una aplicación e incluso repositorios de virus para Android.

Para poder robar información de otro dispositivo es tan fácil como crear un servidor anónimo, hay muchos tutoriales dónde se explica cómo hacerlo, ejecutar la aplicación que inyecta el malware previamente rellenando unos campos y listo. Tienes una aplicación infectada que se conectará a tu servidor. Aunque puedes hacerlo de manera más 'artesanal' con la ayuda de la herramienta fundamental que todas comparten.

Sin ir más lejos, el pasado mes Marzo se registraron un total de casi 5 millones de dispositivos Android infectados a través de un troyano que se auto-instalaba en el terminal. Esto se debe a que ya venía preinstalado y en caso de borrarlo, volvía a instalarse sin el consentimiento del usuario.

Por otro lado, las versiones más antiguas de Android son las más desprotegidas frente a posibles ataques maliciosos. Si nos remontamos al informe de Android proporcionado por Google cada mes, vemos que en Octubre, la versión más usada es Android 7.0 Nougat, es decir, una versión que tiene una antigüedad de más de dos años. Gran parte del porcentaje de versiones usadas se lo llevan también 6.0 Marshmallow y 5.0 Lollipop con casi la mitad entre las dos. Versiones con tres y cuatro años de antigüedad. Y estas versiones están más desprotegidas, aunque cuenten con parches de seguridad. Por lo que atacar a usuarios con versiones desfasadas puede ser más fácil debido al conocimiento interno del sistema operativo gracias al tiempo en el mercado.

Este proyecto abordará todas las fases que hay que seguir y cómo funciona cada una de las partes para poderte dar, como producto final, una aplicación que, a simple vista, no es más que una aplicación más, pero que cuenta con un troyano en su interior comunicándose con un servidor.

Con esto se pretende concienciar a las personas para que tengan cuidado de qué instalan y desde dónde. Aunque quizás no puedas confiar ni de tu mejor amigo si la persona atacante usa correctamente la ingeniería social, fundamental para este tipo de infecciones.

2 ALCANCE Y OBJETIVOS DEL PROYECTO

Este proyecto se ha llevado a cabo para demostrar el desconocimiento de la sociedad ante el ascenso de casos de ataques a dispositivos electrónicos. La información se ha convertido en una fuente de dinero y es por esto por lo que los crackers buscan atacar dispositivos donde más información albergamos. Y uno de esos dispositivos es el smartphone que siempre llevamos encima.

Aquí se han analizado algunas de las herramientas más comunes de infección de Android. En concreto infectar una aplicación de confianza con un troyano que posteriormente se ejecute, en segundo plano, en el dispositivo de la víctima. Este proceso, como se explicará con detalle posteriormente. Se usan herramientas ya conocidas y que se usan de forma legítima.

Se analizará en detalle cómo se puede llegar a extraer todo tipo de información de un dispositivo Android. Para usuarios más experimentados como para otros más aficionados. Haciendo una búsqueda por internet no es difícil topar con algún tutorial para infectar una aplicación con un Android RAT (Android Remote Access Tool).

El objetivo de este proyecto es abrir los ojos a los millones de usuarios que no se preocupan de lo que instalan y lo que aceptan a la hora de instalar una aplicación en su dispositivo. Mostrando la facilidad con la que se puede infectar un dispositivo y cómo los crackers, a partir de programas y de la aplicación de la ingeniería social nos roban toda nuestra información, concienciar a los usuarios.

En este proyecto se enseñará a cómo evitar la infección de nuestros dispositivos más allá de instalar un antivirus. Conocer el panorama y saber cómo actúan los crackers hará que puedas evitar casi cualquier tipo de infección.

3 ESTADO DEL ARTE

Los ataques a dispositivos móviles están a la orden del día gracias a las facilidades que podemos encontrar en internet para elaborar nosotros mismos una aplicación infectada. Es cierto que este tipo de ataques funcionan mejor en versiones no tan nuevas de Android, pero es que debido a la fragmentación que existe con este sistema operativo, es fácil encontrar un dispositivo desactualizado.

Hoy en día, la ejecución de un RAT en un dispositivo con un sistema operativo posterior a Android 7.0 Nougat es mucho más difícil gracias a los parches de seguridad lanzados por Google. Pero como podemos ver en el informe mensual de presencia de sus sistemas operativos en el mercado, más de la mitad de los dispositivos son vulnerables a este tipo de malware.

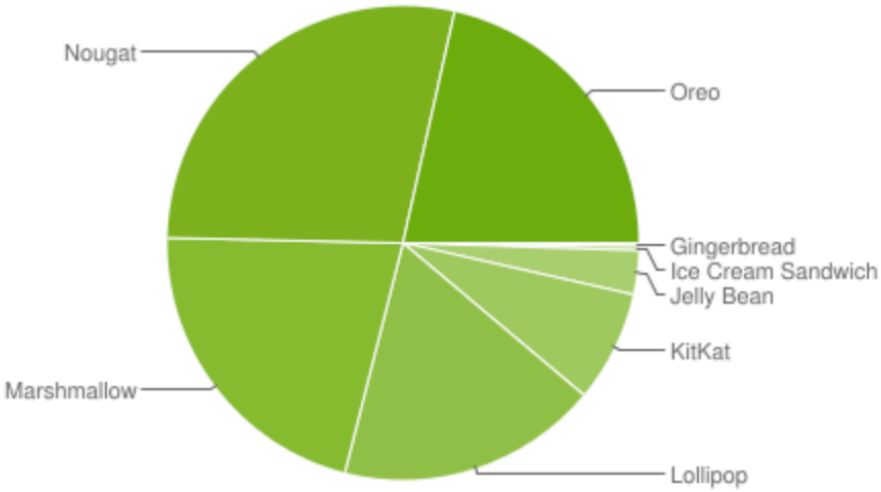

Ilustración 1: Informe de uso de Android en Marzo

Los dispositivos lanzados hoy en día ya están protegidos contra este tipo de ataques. Sin embargo, los crackers siguen y seguirán buscando el método de acceder a tu dispositivo sin tu consentimiento con el fin de sustraer información sensible. Así que es muy importante que la sociedad conozca este problema que afecta a tantas personas a diario y con las que se roban grandes cantidades de dinero.

3.1 Estadísticas

Tan en auge están las aplicaciones maliciosas para Android, que un estudio de la famosa consultora informática alemana G Data afirma que en 2018 creció sobre un 40% en el tercer cuatrimestre del año. Esto se traduce en más de 3,2 millones de aplicaciones con software malicioso que se han registrado.

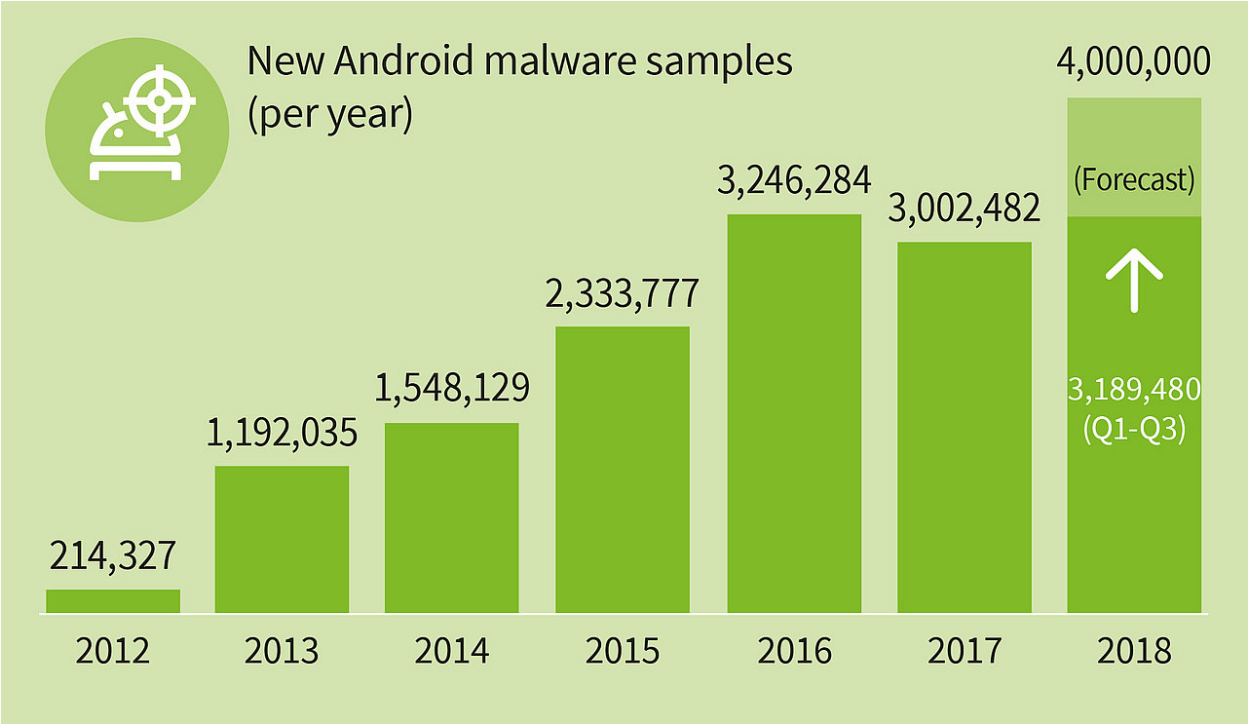

Ilustración 2: Estudio de G Data en 2018

Este estudio por parte de la consultora informática G Data confirma lo que era previsible. Si un sistema operativo es ampliamente usado en todo el mundo, es lógico que los ciberdelincuentes apunten sus armas a dicho sistema operativo. Concretamente, alrededor del 80% de la población mundial con dispositivo móvil, cuenta con un dispositivo con el sistema operativo de Google, Android, en su interior.

Más de cuatro millones de aplicaciones infectadas en 2018 es una cifra que asusta y que debería hacer a los usuarios cuidar más dónde acceden, qué tipo de información consumen y qué aplicaciones instalan.

No acaba aquí, en 2019 se han reportado multitud de ataques al sistema operativo de Google. Fue en Mayo de este año cuando se encontró un sistema de ataque que usaba el sensor de calibrado del dispositivo para inyectar software malicioso tanto en Android como en iOS.

En Marzo de 2019 se reportó una aplicación Android que infectó a más de 250 millones de usuarios. La aplicación, SimBad, se centraba en mostrar anuncios, phishing (suplantación de identidad para la obtención de información sensible) y acceso a información de otras aplicaciones. No sólo eso, sino que permitía abrir e instalar aplicaciones de la Google Play Store sin consentimiento de la víctima, por lo que se convierte en la puerta de acceso perfecta para otros ciberdelincuentes.

Fue en Febrero de 2019 cuando se descubrió una vulnerabilidad de Android donde, haciendo uso de una imagen en formato PNG, se podía infectar un dispositivo. Un exploit que afectaba a las versiónes Android comprendidas entre la 7.0 y la 9.0. Esta imagen puede abrir la puerta a los ciberdelincuentes para ejecutar código malicioso en el dispositivo.

No sólo existen troyanos para Android, sino que hay una técnica muy famosa dentro del mundo del hacking llamada "Man-in-the-Middle" (MITM). Consiste en actuar de interceptor para, mediante Phishing, sustraer información a la víctima que, literalmente, te la da. Este tipo de ataques también son muy comunes y Google trabaja en ello para prevenirlos en Android.

Por último, una de las aplicaciones más usadas en el mundo, WhatsApp, también se utilizó como mecanismo de entrada para la inyección de software malicioso en Android a través de una vulnerabilidad. Esto no es culpa directa del usuario ya que puede afectar a cualquiera.

3.2 Foros de hacking y no hacking

En internet podemos encontrar todo tipo de información con la búsqueda correcta. Y es aquí donde podemos encontrar todo tipo de información para la inyección de software malicioso en una aplicación legítima y consejos para conseguir instalar dicha aplicación en el dispositivo de la víctima.

En una búsqueda usando Google de dos horas ya puedes saber lo básico para empezar a interiorizar en el ámbito de la ciberdelincuencia. Y en un día o dos, ya puedes tener tu aplicación infectada y lista para ser enviada a una persona.

Existen muchas páginas web, foros y documentación sobre la infección de Android con fines académicos, pero esta información se puede usar con otros fines no tan buenos. Durante la documentación para la realización del proyecto se visitaron numerosos foros reconocidos como Reddit y páginas web centradas en la ciberseguridad que albergaban hasta un repositorio con virus listos para usar. Todas estas fuentes se usan para que los interesados prueben dichos ataques en sus propios dispositivos y conocer más sobre la seguridad de su Android. Pero no todos buscan lo mismo.

Hay muchos foros de ciberseguridad como "*elhacker.net*", "*pacientecero.wordpress.com*", "*www.indetectables.net*" entre otros que contienen muchísima información sobre cómo infectar un dispositivo Android. Es por eso que cualquier persona con un mínimo de conocimientos puede, en más o menos tiempo, tener una aplicación infectada con la que sustraer información a otros.

En mi caso, para este proyecto, he ido indagando por multitud de páginas y foros para recabar información. En algunos estaba cierta modificación que necesitaba y en otros alguna herramienta útil, como es el caso de *SignAPK* que se verá después de qué se trata. Pero buscando por internet, se corrobora que puedes tener una aplicación Android infectada con un troyano que se comunique con un servidor.

3.3 Ingeniería social

3.3.1 ¿Qué es la ingeniería social?

La ingeniería social es la práctica de obtener información confidencial a través de la manipulación de usuarios legítimos. Es una técnica que pueden usar ciertas personas para obtener información, acceso o privilegios en un sistema de información que les permitan realizar algún acto que perjudique o exponga la persona u organismo comprometido a riesgo o abusos.

En la práctica, un ingeniero social usará comúnmente el teléfono o Internet para engañar a la gente, fingiendo ser, por ejemplo, un empleado de algún banco u otra empresa, un compañero de trabajo, etc. Esto se le conoce como "pretexto". Ponerte en la piel de otra persona con el fin de camuflar y asegurar el ataque "social" que se pretende hacer para la obtención de esos privilegios o información. Ya ahondaremos posteriormente en los pretextos y en otros aspectos fundamentales que usan los crackers para la infección de los dispositivos.

Aplicar una buena ingeniería social es fundamental para estos crackers ya que, de otro modo, no se podrían obtener los datos mediante la infección de un dispositivo Android. Esto se debe, a que necesitan que la persona afectada sea, ella misma, la que instale el software malicioso en su propio teléfono. Es el trabajo de los crackers hacer eso para poder recopilar la información posteriormente.

Es por esto que es trabajo de todos los usuarios informarse al respecto y de cómo detectar y evitar ataques de ingeniería social. Sospechar de sitios no legítimos y cuidar lo que entra en nuestro dispositivo.

Ya que muchos crackers consideran que el eslabón más débil en la cadena de la seguridad de una empresa es el factor humano, son muchas empresas las que mejoran la seguridad en este aspecto. Dando píldoras formativas a los empleados para evitar este tipo de ataques y con políticas de empresa

muy rigurosas. Aún así, hay personas, auditores, que se encargan de probar la seguridad de las empresas en cuanto al factor humano se refiere. Es el caso de Christopher Hadnagy, autor del libro "Ingeniería Social, el arte del hacking personal", del cual nos basaremos a la hora de hablar de ingeniería social.

3.3.2 ¿Qué aspectos se aplican en la infección con malware?

3.3.2.1 Pretexto

Cómo podemos observar en el libro "Ingeniería Social, el arte del hacking personal", el pretexto se define como la historia, vestimenta, aspecto, personalidad y actitud de fondo que crean la persona que se interpretará durante un ataque. Se engloba todo lo que se pueda imaginar sobre lo que esa persona es. Cuanto más sólido y más simple sea el pretexto, más creíble resultará. Se define como el acto de crear un escenario inventado para incitar al objetivo a que proporcione cierta información o a que realice cierta acción.

El pretexto es un factor muy importante a la hora de infectar un dispositivo con la aplicación troyanizada. El cracker ha de conseguir ganarse la confianza de la víctima para que sea esa persona la que instale, por sí misma, la aplicación infectada.

Para la infección con la aplicación troyanizada en una víctima se usan varios pretextos que son fácilmente identificables:

El propio malware usa un pretexto que es la aplicación en la que va camuflado. En las pruebas realizadas se ha comprobado que se puede infectar desde una aplicación no muy usada como un lector de códigos QR, hasta la aplicación Facebook Lite. Todas ellas con el mismo mecanismo de infección. Este sería un pretexto que se le aplica al propio malware.

Un claro ejemplo de hacer uso de un pretexto, lo que sería suplantación de identidad si nos hacemos pasar por algún organismo o persona conocido, es en el phishing. El phishing, conocido como suplantación de identidad, es un término informático que denomina un modelo de abuso informático y que se comete mediante el uso de un tipo de ingeniería social, caracterizado por intentar adquirir información confidencial de forma fraudulenta (como puede ser una contraseña, información detallada sobre tarjetas de crédito u otra información bancaria).

Durante las pruebas he comprobado que, en mis círculos de amistades y familiares, no es difícil hacer que instalen una aplicación en concreto en sus terminales. Sin embargo, atacar a un completo desconocido con un enlace para descargar una aplicación no sería efectivo.

El propio cracker deberá actuar y aplicarse un pretexto para conseguir ganarse la confianza de la víctima. Si se quiere atacar concretamente a una persona, como puede ser el caso de los robos de información a famosos, los crackers deberán hacer uso de un pretexto con el que ganarse la confianza de la víctima. De este modo pueden conseguir que instales su aplicación en tu terminal y substraer toda la información que se puede recolectar.

Crear este pretexto puede conllevar un periodo de trabajo previo al ataque. Se necesita recolectar información acerca de la víctima o víctimas que se quiera atacar. Para ello, los crackers hacen uso de las redes sociales donde hay una gran cantidad de información acerca de las personas. Es por eso que son muchos los ingenieros sociales y hackers, los que recomiendan cuidar la información nuestra que tenemos en internet y que es accesible a otros.

3.3.2.2 Carga previa

La carga previa consiste en cargar por anticipado a sus objetivos con información o ideas sobre cómo quiere que reaccionen ante cierta información. Este tipo de carga se utiliza comúnmente en la publicidad, en marketing. Se busca implantar las ganas de comprar o de probar algo en la persona para que dicha persona consuma el producto. Y la carga previa es también aplicable, y ha de aplicarse

correctamente si se pretende infectar un dispositivo de una víctima concreta.

La carga previa puede aplicarse a la ingeniería social ya que se busca implantar una idea o pensamiento sin que resulte obvio o avasallador para 'plantar la semilla' en la víctima. Se trata de información que luego la víctima procesará cuando se busque el objetivo final del ataque.

Un ejemplo de carga previa para la infección de una aplicación infectada sería recomendar ese tipo de aplicaciones, no la tuya propia. Por ejemplo, si lo ocultaste detrás de un gestor de tareas, decir que has usado gestores de tarea y que te ayudan a organizarte. Si esa persona es organizada, seguro que puedes hacer que instale tu aplicación y fin del ataque.

3.3.3 Trabajo previo a la infección con malware

La infección de un dispositivo necesita de trabajo previo que un cracker tendrá que realizar si se quiere infectar a un individuo o grupo concreto. No todo es la creación de la aplicación infectada, sino que se requiere de cierta ingeniería social para llevar a cabo un ataque eficiente.

Una aplicación troyanizada necesita de ingeniería social o aprovechar un momento de auge de alguna aplicación aún no disponible en Google Play para funcionar. Como experiencia personal, hubiera sido fácil infectar una buena cantidad de móviles ante la llegada de una novedad a Whatsapp que sólo estaba disponible en la fase beta y para ello se necesita instalar un APK descargado. Esos momentos son los que más atentos hay que estar con lo que se instala y siempre optar por instalar desde Google Play.

3.3.4 La seguridad con respecto a la ingeniería social es también importante

La ingeniería social es un aspecto de la seguridad de la información que también cuidan las empresas. Existen compañías que realizan auditorias de ingeniería social a las empresas con el fin de recabar información sobre puntos de mejoras y análisis del estado actual de la empresa frente a un ataque de este tipo.

Los tipos de ataque que se pueden realizar pueden ser:

- Envío de correos y/o ficheros. Este método consiste en mandar un correo electrónico que posteriormente abra la víctima e interactúe con él.
- Puntos de acceso WiFi. Suplantación (o creación de uno nuevo) del punto de acceso WiFi de la compañía. Con esto se busca conseguir que la víctima se conecte a esta red y poder extraer la información.
- Clonación de webs. Clonar una web a la que la víctima suela acceder para poder extraer las credenciales.
- Mensajería instantánea y SMS. Al igual que el envío por correo electrónico, los servicios de mensajería instantánea también se atacan con el fin de extraer información. Servicios como Whatsapp y Skype.
- Distribución de USB. Se comprueba la facilidad con la que poder infectar un dispositivo interno con un USB infectado abandonado intencionadamente dentro o en los alrededores de la empresa.
- Códigos QR. Se pretende infectar el dispositivo de la víctima mediante un papel publicitario con un código QR impreso en él.
- Búsquedas en redes sociales de información publicada por los empleados de la empresa
- Llamadas telefónicas con suplantación de identidad para la extracción de información
- Acceso físico a instalaciones validando controles de acceso desde el exterior y el interior

- Ataques de phishing (El phishing es un método que los ciberdelincuentes utilizan para engañar y conseguir que se revele información personal, como contraseñas o datos de tarjetas de crédito y de la seguridad social y números de cuentas bancarias)

Con la evolución de la tecnología y de las medidas técnicas de seguridad, si algo ha quedado patente es que el eslabón más importante de la famosa cadena de la seguridad de la información de las organizaciones es la persona. La seguridad de la información no se garantiza únicamente con la definición de procedimientos de gestión de incidencias o la implantación de mecanismos de cifrado, por poner algunos ejemplos.

Por ese motivo, es fundamental que el personal de la organización esté concienciado e implicado. No sólo en el cumplimiento de las normas que se hayan implantado, sino también manteniendo una actitud de precaución y alerta en el uso cotidiano de los sistemas de información, en las relaciones personales y laborales. Los empleados son la última barrera del sistema de defensa de la seguridad de las empresas.

Porque somos personas, somos seres humanos. En eso se basa la ingeniería social: en el principio de que el punto más débil de cualquier sistema de defensa son las personas.

Los ciberdelincuentes se han dado cuenta de que tienen que diseñar acciones orientadas a intentar hacer vulnerable esa última línea de defensa y han decidido abrir vías de ataque que apuntan directamente a ese eslabón tan importante de la cadena: el empleado.

Actualmente la ingeniería social es uno de los vectores de ataque más peligrosos y que más se está utilizando para acceder a las redes de las organizaciones, haciendo uso de los empleados de las propias organizaciones.

La ingeniería social se compone de técnicas de engaño de todo tipo, tanto en el mundo físico como en el virtual, y se basa en tres principios básicos de las relaciones humanas:

- El primer movimiento es siempre de confianza hacia el otro.
- Todos queremos ayudar.
- No nos gusta decir No.

Así que es muy común toparse con algún caso de ataque hacia la persona para obtener cierta información. Si ocurre en las grandes empresas donde más seguridad tienen, ¿por qué no iban a atacar a un usuario medio de Android con el fin de amenazarle posteriormente tras recabar información sensible?

Además, viendo cómo es de sencillo troyanizar una aplicación para que parezca inofensiva, ser víctima de un ataque es más posible de lo que se piensa. Por eso hay que tener cuidado en quién se confía y la fuente de la que se descargan las aplicaciones que instalamos en nuestro dispositivo.

4 MARCO DE REFERENCIA:

En el mundo de la ciberseguridad existen muchos mecanismos de infección en dispositivos electrónicos. Los sistemas operativos, aunque luchan a diario por ser más robustos, siempre estarán al acecho del ataque de crackers que busquen vulnerabilidades con las que inyectar código malicioso.

4.1 Qué es un troyano

En este proyecto se ha abordado el tipo de inyección mediante troyano. Un caballo de Troya o troyano es un tipo de malware que a menudo se camufla como software legítimo. Los ciberladrones y los hackers pueden emplear los troyanos para intentar acceder a los sistemas de los usuarios. Normalmente, algún tipo de ingeniería social engaña a los usuarios para que carguen y ejecuten los troyanos en sus sistemas.

Un troyano es un sistema de infección de dispositivos que siempre está evolucionando para evitar los bloqueos que ponen los fabricantes. Por ello, nunca se estará protegido al completo de una posible infección. Dentro del mundo de las telecomunicaciones, donde todo está interconectado gracias a internet, propagar un software malicioso es considerablemente fácil en comparación con los mecanismos de robo de datos de antaño.

Existen varios sistemas de detección de troyanos en el mercado, mediante librerías de troyanos previamente detectados, se puede detectar uno nuevo si hace uso de software previamente encontrado. Dendroid se basa en AndroRAT, un troyano para Android que ya ha sido detectado y que lo bloquea Google en Android de forma nativa. Pero un grueso de los usuarios no tiene dispositivos actualizados, usan dispositivos antiguos y que sufren más las infecciones mediante troyanos de los ciberdelincuentes.

Personas con menos recursos y conocimientos que son más vulnerables a una extorsión tras una extracción de información sensible de su dispositivo. Existen troyanos que 'secuestran' tu dispositivo cifrándolo y que roban todas tus cuentas. En este caso, AndroRAT puede controlar en segundo plano todo el dispositivo por lo que puede hacer aún más daño en el caso de una infección exitosa.

Con este proyecto se pretende concienciar de la existencia de estos virus en Android que son capaces de hacer muchísimo daño a las personas que sufren una infección de este tipo.

4.2 Funcionamiento de un troyano

Una vez activados, los troyanos pueden permitir a los cibercriminales espiarte, robar tus datos confidenciales y obtener acceso por una puerta trasera a tu sistema. Estas acciones pueden incluir las siguientes:

- Eliminación de datos
- Bloqueo de datos
- Modificación de datos
- Copia de datos
- Interrupción del rendimiento de ordenadores o redes de ordenadores

Debido a la enorme cantidad de información que albergamos en nuestros teléfonos móviles, los troyanos son la herramienta más habitual para el robo de información dentro de estos dispositivos. Por eso, para este proyecto, se ha utilizado de un troyano llamado Dendroid, que se inserta dentro de una aplicación legítima y que se ejecuta posteriormente en segundo plano sin rastro para la víctima.

4.3 Clasificación de los troyanos existentes hoy en día

Los troyanos se clasifican en función del tipo de acciones que pueden realizar en el ordenador:

- **Puerta trasera** (**Backdoor**): Un troyano backdoor (de puerta trasera) proporciona el control remoto del ordenador infectado a los ciberdelincuentes. Estos troyanos permiten al ciberdelincuente hacer todo lo que desee en el ordenador infectado, como enviar, recibir, iniciar y eliminar archivos, mostrar datos y reiniciar el ordenador. Los troyanos backdoor (de puerta trasera) a menudo se utilizan para unir un conjunto de ordenadores infectados para formar un botnet o una red zombi que se puede utilizar con objetivos delictivos.
- **Exploit**: Los exploits son programas que contienen datos o código que se aprovechan de una vulnerabilidad del software de aplicaciones que se ejecuta en el ordenador.
- **Rootkit**: Los rootkits están diseñados para ocultar ciertos objetos o actividades en el sistema. A menudo, su objetivo principal es evitar la detección de programas maliciosos con el fin de ampliar el periodo en el que los programas pueden ejecutarse en un ordenador infectado.
- **Trojan-Banker** (Troyano Bancario): Los programas Trojan-Banker, o troyanos bancarios, están diseñados para robar tus datos bancarios de sistemas de banca online, sistemas de pago electrónico y tarjetas de débito o crédito.
- **Trojan-DDoS**: Estos programas realizan ataques DoS (denegación de servicio) contra una dirección web específica. Mediante el envío de una gran cantidad de solicitudes (desde tu ordenador y otros ordenadores infectados), el ataque puede saturar la dirección de destino y originar una denegación de servicio.
- **Trojan-Downloader**: Los programas de descarga de troyanos, Trojan-Downloader, pueden descargar e instalar nuevas versiones de programas maliciosos en el ordenador, incluidos troyanos y adware.
- **Trojan-Dropper**: Los hackers utilizan estos programas para instalar troyanos y virus, o bien para evitar la detección de programas maliciosos. No todos los programas antivirus pueden analizar todos los componentes incluidos en este tipo de troyano.
- **Trojan-FakeAV**: Los programas Trojan-FakeAV simulan la actividad de software antivirus. Están diseñados para extorsionar al usuario a cambio de la detección y eliminación de amenazas, aunque estas no existan realmente.
- **Trojan-GameThief**: Este tipo de programas roba los datos de la cuenta de usuario de los jugadores online.
- **Trojan-IM**: Los programas Trojan-IM roban los datos de inicio de sesión y las contraseñas de los programas de mensajería instantánea, como ICQ, MSN Messenger, AOL Instant Messenger, Yahoo Pager, Skype, etc.
- **Trojan-Ransom**: Este tipo de troyano puede modificar los datos del ordenador para que no funcione correctamente o no puedas utilizar datos específicos. El cibecriminal solo restaurará el rendimiento del ordenador o desbloqueará los datos una vez que hayas pagado el dinero "de rescate" que solicita.
- **Trojan-SMS**: Estos programas pueden costarte dinero en forma de envío de mensajes desde el dispositivo móvil a números de teléfono con tarificación especial.
- **Trojan-Spy**: Los programas Trojan-Spy pueden espiar cómo utilizas el ordenador. Por ejemplo, mediante el seguimiento de los datos que introduces a través del teclado, la realización de capturas de pantalla o la obtención de una lista de aplicaciones en ejecución.

- **Trojan-Mailfinder**: Estos programas pueden recopilar las direcciones de correo electrónico del ordenador.
- Otros tipos de troyanos:
    - Trojan-ArcBomb
    - Trojan-Clicker
    - Trojan-Notifier
    - Trojan-Proxy
    - Trojan-PSW

4.4 Software malicioso elegido para la elaboración de este Proyecto

En la elaboración de este proyecto, se ha usado un troyano *backdoor*. En eso consiste un AndroRAT, en un proceso que se ejecuta en segundo plano y es capaz de controlar el dispositivo de forma ajena a la víctima.

Dendroid, una adaptación del AndroRAT original de código abierto descubierto por Robin David, tiene la capacidad de controlar, de forma ajena a la víctima, el dispositivo en el que está instalado. Permite al atacante desde hacer fotografías y grabar audio, hasta forzar la apertura de páginas en el navegador de forma repentina. Aprovecha una vulnerabilidad de Android, que ya está más controlada, para permitir al atacante controlar de forma libre el dispositivo infectado.

Se ha elegido este troyano porque es, sin duda, el más completo para infectar un dispositivo Android. Dendroid te permite recopilar información del usuario, como contraseñas y fotografías sensibles, sin el conocimiento de la víctima. Un troyano completo y que es fácilmente adaptable para inyectarse en una aplicación legítima. Además, ofrece un panel de control muy completo y fácilmente instalable con el que controlar todos los dispositivos que se hayan infectado.

5 METODOLOGÍA DE TRABAJO

5.1 Metodología principal

Durante la elaboración del proyecto he seguido un procedimiento de información, implementación y pruebas. Es decir, realizaba una documentación inicial para conocer cómo realizar la implementación. Posteriormente, modificaba los archivos necesarios y generaba una aplicación comprimida en formato APK que posteriormente probaba en un terminal Android físico.

Con cada error lo iba anotando para no volverlo a cometer posteriormente en un bloc de notas. Al igual que lo que conllevaba a avances en la correcta implementación del proyecto.

Como muchos de estos tutoriales e información recopilada en internet, es muy importante recopilar una buena cantidad de información previa a empezar a modificar código o directorios.

5.1.1 Metodología para la documentación

En el proceso de documentación, he investigado en internet ya que es donde es más fácil encontrar información al respecto. Aunque este tipo de implementaciones, como es la infección de una aplicación legítima para el robo de información, se centran en enseñar, siempre habrá quién las use de manera errónea.

5.1.2 Metodología para la implementación

Durante la elaboración del proyecto se han realizado multitud de pruebas para la inyección del troyano según la documentación y tutoriales recopilados por internet. Diferentes métodos para la copia de archivos y la modificación de otros para el correcto funcionamiento de la aplicación recompilada.

En la elaboración del proyecto se han modificado, principalmente, los directorios de las aplicaciones tanto las legítimas como el troyano AndroRAT. Esto es porque la aplicación troyano sólo se necesitaba modificar una vez. Se modificó tanto la dirección IP del servidor donde se ejecutaría como, en nuestro caso, que no dejara rastro con un icono falso en el cajón de aplicaciones.

Una vez se tuvo la aplicación AndroRAT ya configurada haciendo uso de Android Studio, se descompiló con ApkTool y se mantuvo ese directorio. Lo que se modificaban eran los directorios de las otras aplicaciones legítimas que se descompilaban para insertar el virus en ellas y volver a recompilar. Se copiaba, se modificaba el AndroidManifiest.xml y se volvía a compilar. Si fallaba, se buscaba otro método de hacer que compilara. Si compilaba, se comprobaba posteriormente en el terminal Android para ver si instalaba correctamente. Y, por último, si la aplicación funcionaba una vez reiniciado el dispositivo Android. Esto es, porque en el AndroidManifiest.xml se configuró para que la actividad del AndroRAT se iniciase al iniciar el dispositivo.

5.1.3 Metodología para las pruebas

Con el fin de comprobar el correcto funcionamiento de la aplicación se recurrió a la instalación de la aplicación compilada en el dispositivo Android de pruebas del que se disponía. Se había comprobado, previamente, que el AndroRAT se comunicaba perfectamente con el servidor y podíamos controlar el dispositivo de forma remota y ajena al usuario. Lo que se comprobaba a posteriori, era si el AndroRAT introducido en una aplicación legítima seguía funcionando.

Una vez conseguida la ejecución del AndroRAT en segundo plano de forma oculta en el dispositivo, se comprobó que aparecía, efectivamente, en el panel de control que incluye Dendroid. Aquí se hicieron pruebas como la captura de fotografías, abrir páginas web y extraer posteriormente las imágenes capturadas por la cámara frontal. Todo esto de forma ajena al que sería el usuario.

5.2 Software complementario

5.2.1 Telegram

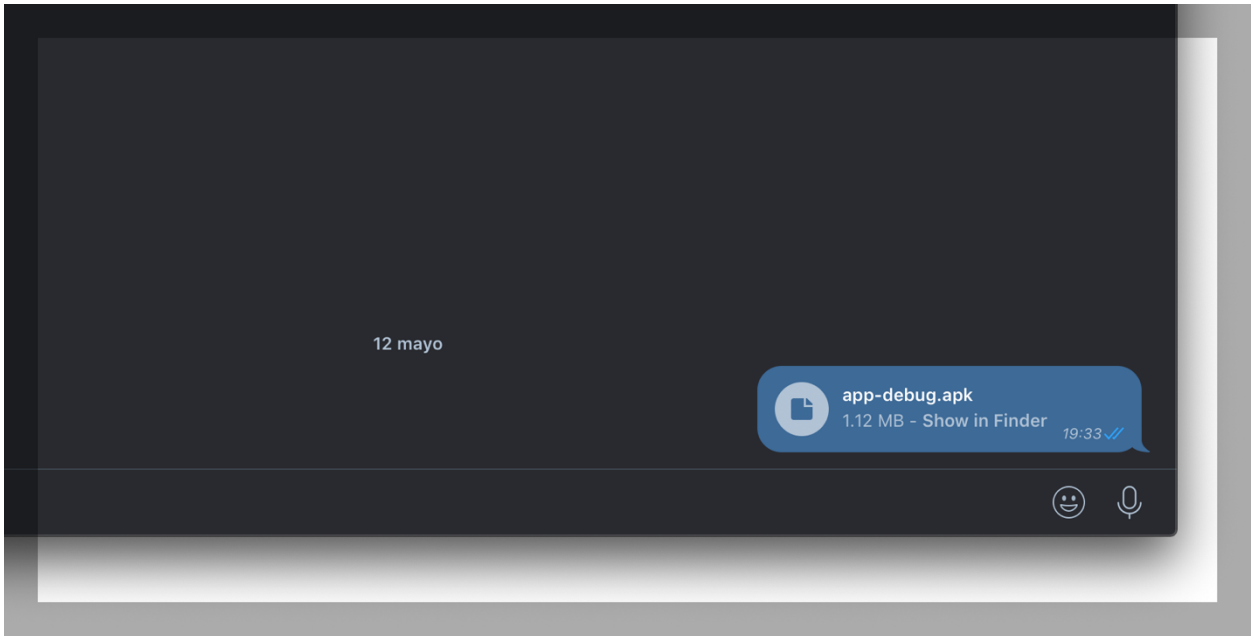

Ilustración 3: Captura de Telegram

Telegram es un servicio de mensajería instantánea ampliamente conocido por millones de usuarios. Está basado en código libre y cualquier persona puede descargarlo para modificarlo. Esta aplicación es multiplataforma y en mi caso ese fue el motivo por el que me dispuse a usarla.

Esto es porque me permitía el paso de información entre mi ordenador y el dispositivo Android que he usado durante la elaboración del proyecto. El principal uso fue comprobar que se podía enviar, recibir e instalar la aplicación infectada mediante la transferencia de dicho archivo por Telegram.

5.2.2 Google Drive

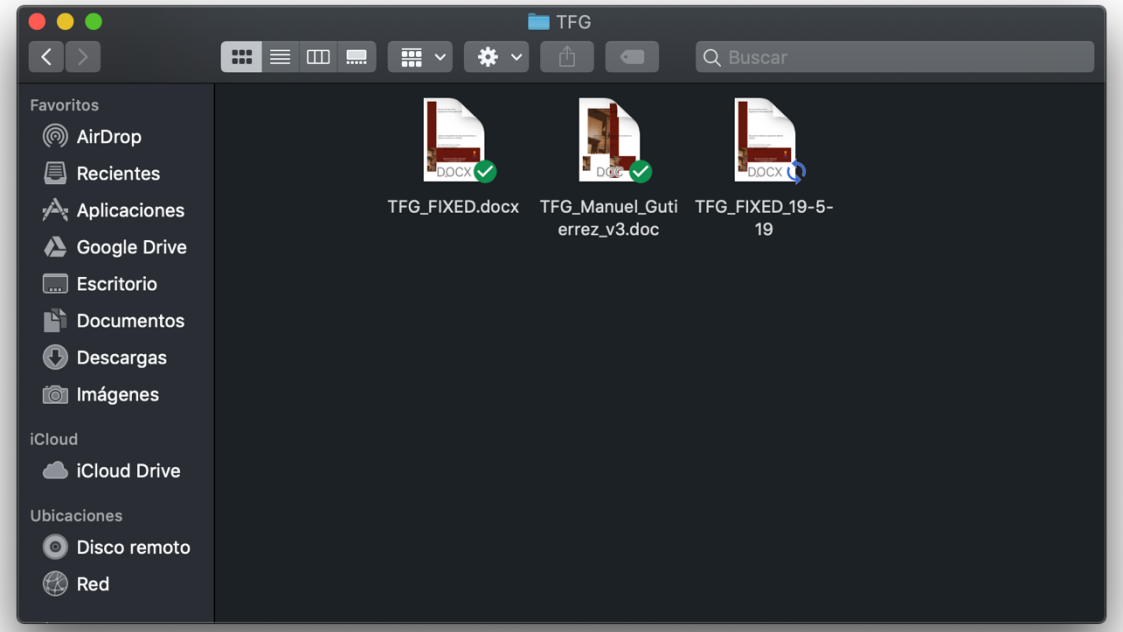

Ilustración 4: Carpeta 1 del proyecto en Google Drive

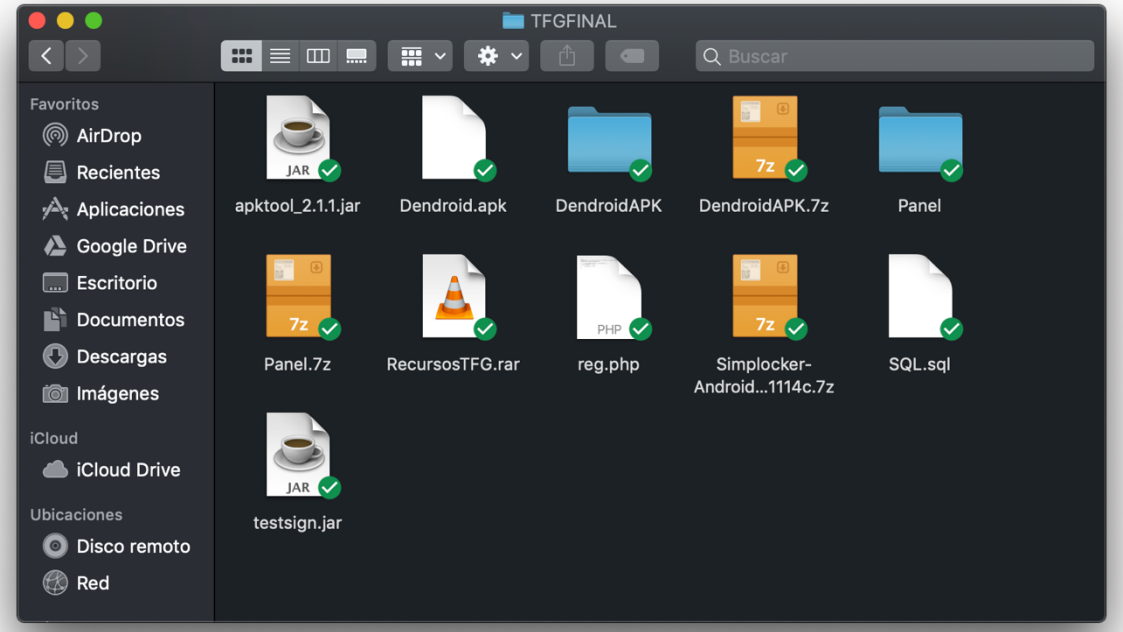

Ilustración 5: Carpeta 2 del proyecto en Google Drive

Google Drive es un servicio de almacenamiento en la nube multiplataforma. Lo he utilizado para almacenar cada una de las versiones del código utilizado, así como respaldar el documento aquí presente. Esto me permitió tener una copia de seguridad en caso de algún tipo de problema con el ordenador y poder acceder a él desde cualquier otro dispositivo.

En Drive he almacenado cada aplicación infectada compilada para poder analizarla posteriormente en la máquina virtual que tenía instalada en Virtual Box. También todas las herramientas todo-en-uno para la generación de una aplicación maliciosa y la documentación importante para el transcurso de este proyecto.

6 IMPLEMENTACIÓN DEL PROYECTO

6.1 Infografía del entorno usado para las pruebas

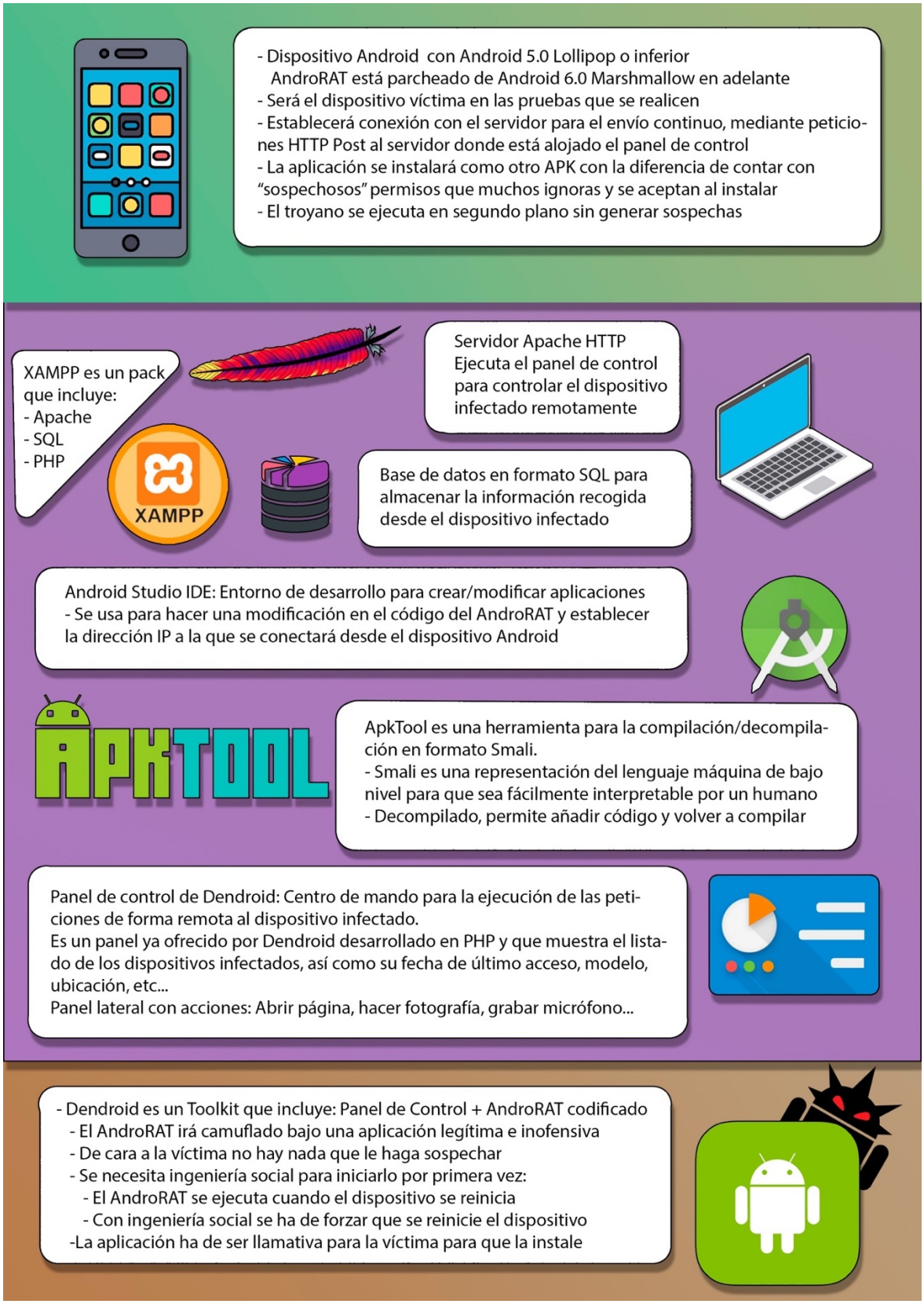

Ilustración 6: Infografía del entorno utilizado

6.2 Herramientas utilizadas

6.2.1 Hardware utilizado

En el desarrollo de las pruebas y de la investigación sobre las herramientas de inyección de código malicioso hice uso de dos ordenadores distintos:

1. **Portátil MacBook Pro**:
    a. Sistema operativo: Mac OS.
    b. Entorno de desarrollo instalado: Android Studio
    c. Servidor Apache y MySQL con XAMPP
    d. Sistema de ofimática para la documentación: Microsoft Word
    e. Sistema de almacenamiento de datos: Google Drive
2. **Dispositivo Android** (Smartphone)
    a. En el desarrollo de este proyecto y a sabiendas que se necesitarían realizar pruebas reales para comprobar el funcionamiento de las diversas herramientas, se hizo uso de un Sony Xperia X Compact. Este dispositivo contaba con la versión de Android 8.1 Oreo y contaba con aplicaciones básicas instaladas.
    b. Dispositivo Motorola Moto G de primera generación con Android 5.0 para poder comprobar el correcto funcionamiento del proyecto debido a las limitaciones de seguridad impuestas por versiones posteriores.

6.2.2 Software utilizado

6.2.2.1 Android Studio

Android Studio es el entorno de desarrollo integrado oficial para la plataforma Android. Desarrollado por Google y creador por IntelliJ para facilitar la creación de aplicaciones a los desarrolladores. Un entorno muy potente que me ha permitido modificar el software malicioso que luego se inyectaría a la aplicación que queramos infectar.

Está basado en el software IntelliJ IDEA de JetBrains y está disponible de forma gratuita bajo la licencia Apache 2.0. Un entorno de desarrollo centrado para facilitar el uso del SDK de Android y la creación de nuevas aplicaciones.

6.2.2.2 Apktool

Esta herramienta es un software para realizar ingeniería inversa a aplicaciones Android en formato APK. Es capaz de decodificar el archivo APK en ficheros legibles y modificables por un usuario. Permite, además, la posibilidad de recompilar la aplicación tras una modificación interna. Esto ha sido de gran ayuda para poder introducir el código malicioso dentro de una aplicación legítima. Es la herramienta que comparten todos los sistemas de inyección de malware que se han probado.

6.2.2.3 ADB

Android Debug Bridge (ADB) es una herramienta de líneas de comandos versátil que te permite comunicarte con una instancia de un emulador o un dispositivo Android conectado. Esta herramienta proporciona diferentes acciones en dispositivos, como la instalación y la depuración de apps, y proporciona acceso a un shell Unix que puedes usar para ejecutar varios comandos en un emulador o un dispositivo conectado. Es un programa cliente-servidor que incluye tres componentes:

- Un cliente, que envía comandos. El cliente se ejecuta en tu máquina de desarrollo. Puedes invocar un cliente desde un terminal de línea de comandos emitiendo un comando de ADB.
- Un daemon, que ejecuta comandos en un dispositivo. El daemon se ejecuta como un proceso en segundo plano en cada instancia del emulador o dispositivo.
- Un servidor, que administra la comunicación entre el cliente y el daemon. El servidor se ejecuta como un proceso en segundo plano en tu máquina de desarrollo.

6.2.2.4 XAMPP

XAMPP es un paquete de software libre, que consiste principalmente en el sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script PHP y Perl. El nombre es en realidad un acrónimo: X (para cualquiera de los diferentes sistemas operativos), Apache, MariaDB/MySQL, PHP, Perl. A partir de la versión 5.6.15, XAMPP cambió la base de datos MySQL por MariaDB, un fork de MySQL con licencia GPL.

El programa se distribuye con la licencia GNU y actúa como un servidor web libre, fácil de usar y capaz de interpretar páginas dinámicas. A esta fecha, XAMPP está disponible para Microsoft Windows, GNU/Linux, Solaris y Mac OS X.

6.3 Preparación del entorno

En la elaboración de este proyecto se han usado varios entornos y programas de fácil acceso para cualquier usuario. Se ha utilizado un ordenador portátil y herramientas gratuitas que se pueden encontrar por internet y que, la mayoría, vienen ya recopiladas en tutoriales de foros y blogs en internet.

Con esto se pretende concienciar a todo el mundo que no se necesitan demasiados conocimientos ni dinero para poder tener en tu poder una aplicación que pueda infectar un dispositivo Android y sea capaz de extraer información.

A continuación, vamos a detallar el procedimiento realizado para la preparación del entorno de pruebas y las herramientas utilizadas en los dos casos que se han abordado. El primer caso es un método más purista para personas con conocimientos sobre la programación y la tecnología. Y un segundo caso donde se utiliza una herramienta guiada que te arroja una aplicación en formato APK ya infectada y lista para usarse.

6.3.1 Instalación del entorno de pruebas

El entorno de pruebas que se ha utilizado ha sido un dispositivo Android, el Motorola Moto G de primera generación con Android 5.0 Lollipop instalado. Este sistema operativo sigue presente en gran parte de dispositivos móviles y puede ser el foco de ataque de muchos crackers. Dicho dispositivo es el que se ha estado utilizando durante las pruebas para la instalación del software malicioso y comprobar su respuesta, si llegaba a conectar con el servidor y si había comunicación entre servidor y dispositivo infectado.

Por otro lado, el ordenador portátil se ha utilizado como servidor Apache con MySQL incorporado y compatibilidad con PHP para la ejecución del servidor. Los ficheros para el servidor se pueden encontrar fácilmente en internet y su instalación es muy sencilla si se hace uso, por ejemplo, de XAMPP, un entorno que ya lo engloba todo en una sola instalación.

6.3.1.1 Instalación de XAMPP (Apache y MySQL)

XAMPP es un paquete instalable que incluye tanto un servidor de Apache como una base de datos basada en MySQL. El motivo de usar este paquete es que facilita enormemente la preparación del entorno ya que, además de incluir el servidor y la base de datos, viene ya preparado para ser compatible con ficheros en PHP. Esto hace que preparar el servidor no nos lleve más de media hora en dejarlo ya funcionando. Sólo hay que cambiar ciertas configuraciones que luego se detallarán.

6.3.1.2 Instalación de Android Studio

El proyecto donde viene desarrollado el AndroRAT Dendroid se puede abrir tanto con Eclipse, IntelliJ o Android Studio. Dicho código ha de modificarse para que haya comunicación entre el dispositivo infectado y el servidor que estamos ejecutando en el ordenador. En este proyecto he decidido usar Android Studio ya que, aunque el código es Java, Android Studio viene más preparado para comunicarse con el dispositivo y agilizaba el proceso de pruebas.

Para la instalación de este entorno de desarrollo, se ha descargado el instalador desde la página oficial de Google y se ha instalado como un programa más.

6.3.2 Preparación de las herramientas

6.3.2.1.1 Preparación de los archivos necesarios – Descargas, ficheros, carpetas, instaladores

En primer lugar, tendremos que descargar los programas y herramientas que serán necesarias para la implementación de este proyecto. Concretamente, el método manual que se va a explicar a continuación.

Para ello descargaremos los siguientes archivos:

- Dendroid
- Android Studio IDE
- ApkTool
- SignAPK (JarSigner en su defecto)
- Xampp
- MySQL
- Editor de texto alternativo o el por defecto del sistema operativo usado
- Java SE Development Kit 8

Con estas herramientas ya podremos tener una aplicación Android infectada con un AndroRAT en su interior. Aclarar que todas son gratuitas y fáciles de encontrar en internet.

6.3.2.1.2 Preparación del servidor HTTP Apache

La preparación del servidor HTTP Apache es una de las partes más importantes para la correcta implementación del proyecto. La aplicación se conecta a dicho servidor para el envío y recepción de información.

La instalación del servidor HTTP Apache, puesto que se necesitan también los módulos para lenguaje PHP, se realiza a través de la herramienta XAMPP. Esta herramienta es una distribución de Apache completamente gratuita y fácil de instalar que contiene MariaDB, PHP y Perl. El propio paquete de instalación te permite elegir qué instalar, en el caso de este proyecto, se eligió el servidor Apache HTTP y PHP.

Cuando se ha instalado el servidor Apache, es necesario hacer ciertas modificaciones previas en los ficheros de configuración. Para ello en la propia herramienta XAMPP Control Panel se puede acceder a todos los ficheros que se han de modificar.

En primer lugar, se ha de modificar el fichero "*httpd.conf*" para que se pueda desplegar el servidor en una dirección IP o sub-dominio concreto. Para ello abrimos el fichero "*httpd.conf*" con el editor de texto que se quiera, durante la implementación se hizo con el propio Bloc de Notas de Windows 10 o con Notepad++.

Primero tendremos que modificar el "*ServerName*". En mi caso se estableció una IP fija para el ordenador donde se había instalado el servidor y esa era la dirección IP del servidor.

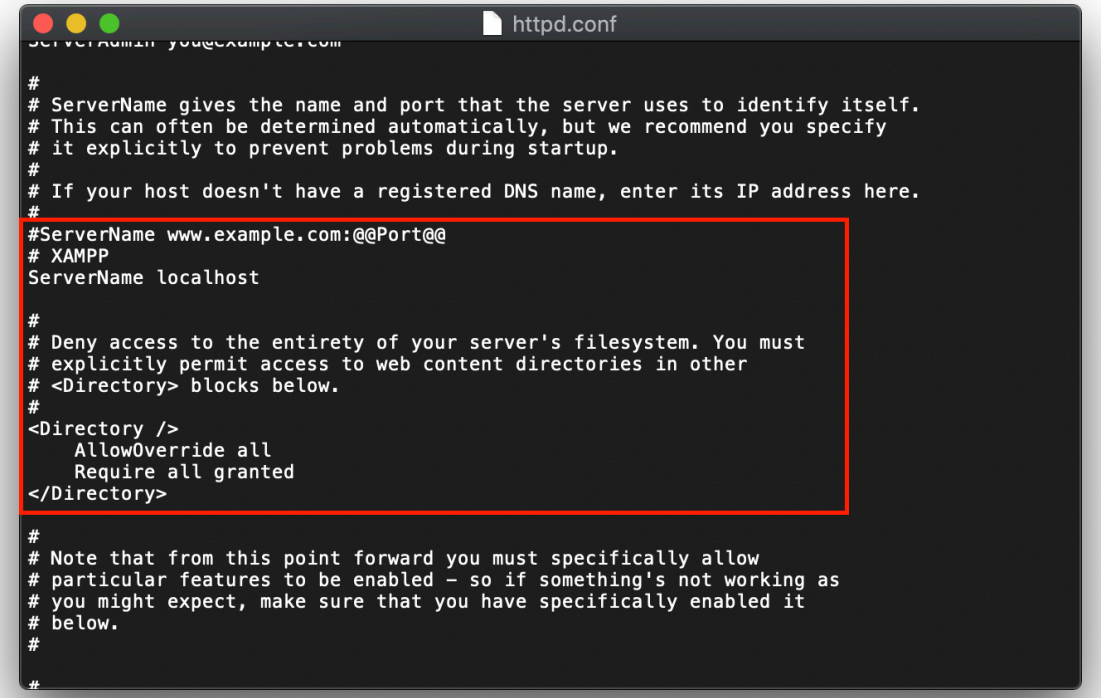

Ilustración 7: Modificación httpd.conf para el servidor Apache

En internet, buscando información sobre Dendoroid, me topé con muchos tutoriales que explicaban a la perfección cómo crear un subdominio "*no-ip*" y cómo configurar luego tu router para poder recibir las peticiones a tu servidor de manera enmascarada.

Posteriormente se necesita modificar una propiedad referente al directorio donde se alberga el Panel ya preconfigurado en lenguaje PHP. Para ello hay que modificar el apartado "*Directory*" en el fichero "*httpd.conf*".

Es necesario configurar el directorio dónde se almacenará el Panel que servirá de puente entre la aplicación infectada en el dispositivo de la víctima y nosotros. En mi caso no he modificado el directorio y dejé el valor por defecto.

Por último, se ha de modificar el fichero "*reg.php*" dentro de la carpeta Panel antes de copiarla al directorio seleccionado en "*DocumentRoot*". La carpeta Panel se copia en dentro del directorio y para acceder al panel se hará "*ServerName/Panel*".

Dentro del fichero "*reg.php*" se especifican los dominios aceptados por el panel de control. En las pruebas se usó la dirección IP del ordenador, por lo que el fichero "*reg.php*" queda así:

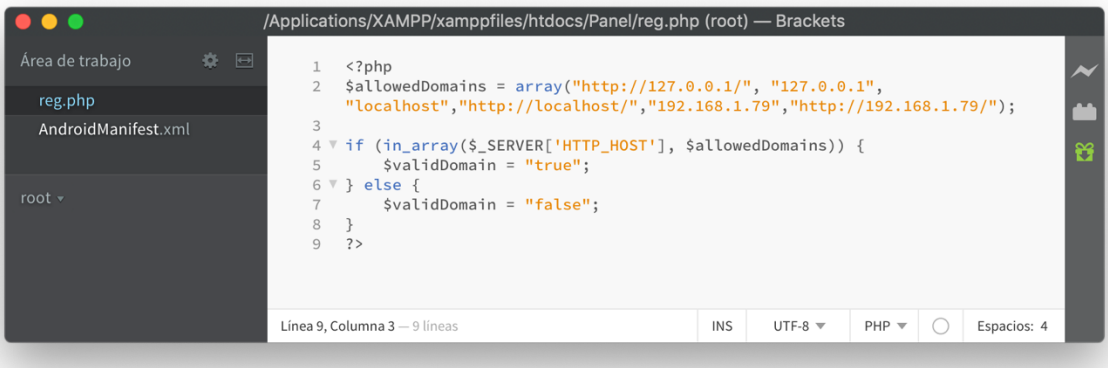

Ilustración 8: Modificación del reg.php para acceder al panel de Dendroid

A continuación, se ha de otorgar permisos de ejecución a la carpeta que incluye el panel de control del AndroRAT. Para ello tendremos que hacer uso de la ventana de comandos, en mi caso el Terminal de Mac OS, ir hacia la carpeta */htdocs* y hacer uso de la herramienta *CHMOD* para cambiar los permisos.

Hecho todo este procedimiento, abrimos el navegador que usemos habitualmente, en mi caso Google Chrome, y se accede a la dirección IP que hemos configurado en el fichero **httpd.conf**. Esto nos redigirá a una página de acceso donde tendremos que ingresar el siguiente usuario/contraseña:

```
Usuario/Contraseña: master/Dam34
```

Pero este panel de control no se mostrará correctamente hasta que no se configure la base de datos, que será la que recopilará y permitirá mostrar la información a través del panel de control.

6.3.2.1.3 Preparación de la base de datos MySQL

Este panel requiere de una base de datos SQL que permita organizar tanto el panel como la información que recopile de los dispositivos infectados. Dendroid viene ya con un fichero en formato SQL que viene preparado para cargarlo en el programa MySQL sin necesidad de tocar nada más.

El fichero SQL que incluye Dendroid se ha cargado haciendo uso de **PhpMyAdmin**, una herramienta que se incluye en el paquete de XAMPP. Importar un fichero .SQL consiste en pulsar sobre la pestaña de importar, seleccionar el archivo y dejar que **PhpMyAdmin** haga todo el proceso de configuración de la base de datos Dendroid.

Por defecto, MySQL habilita un modo de tabla SQL que no es compatible con el fichero de configuración ofrecido por Dendroid. Por ello, tenemos que modificar el fichero **my.conf** de la base de datos añadiendo la línea marcada en la siguiente captura:

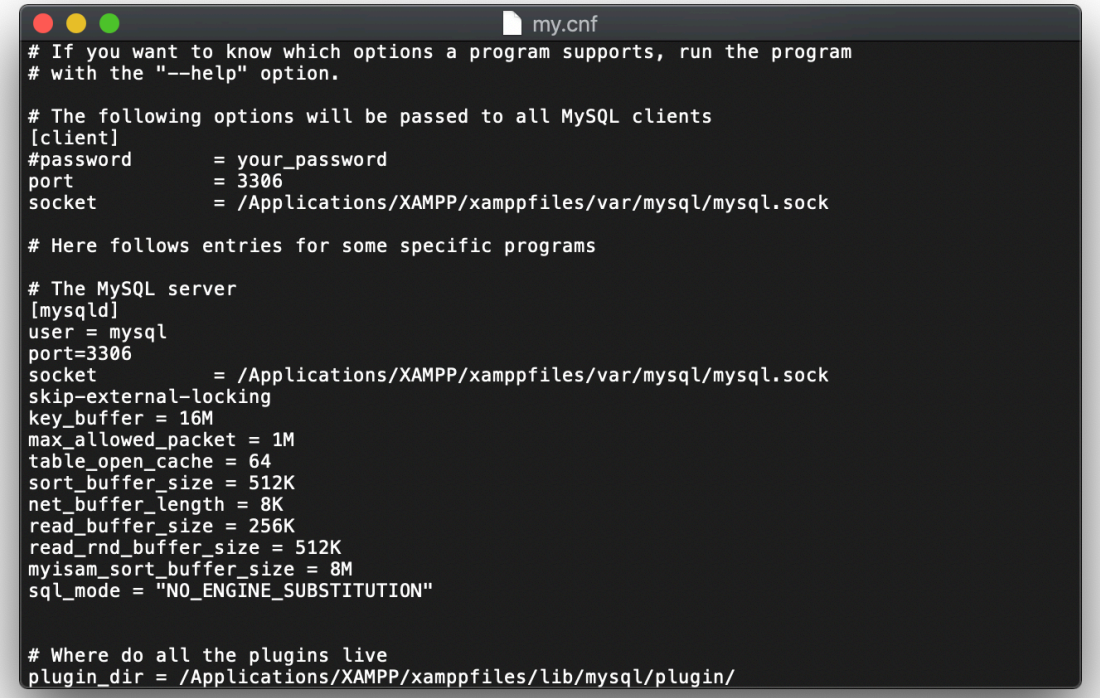

Ilustración 9: Modificación my.conf de la base de datos

Para acceder a este fichero de configuración se ha de abrir el panel de control de XAMPP, pulsar sobre Manage Servers y pulsando previamente sobre el servicio MySQL, pulsamos después en Configure. Esto nos permitirá abrir el fichero my.conf y añadir la línea *"sql_mode = NO_ENGINE_SUBSTITUTION"* que es fundamental para el correcto funcionamiento del panel.

Esta configuración es muy importante para el correcto funcionamiento del panel de control de Dendroid. En caso de que no cargue dicho panel, en mi caso, fue este el principal problema.

Con estos cambios hechos en la configuración del servidor Apache y en la base de datos, el panel de control de Dendroid debería cargar perfectamente.

6.3.2.1.4 Preparación de Android Studio

Para preparar Android Studio sólo hay que instalar mediante el procedimiento guiado que incluye. Se necesitará este IDE para poder abrir el proyecto de Dendroid y modificar ciertos archivos incluidos en él. Una vez modificados los archivos, se utilizará Android Studio para exportar la aplicación en formato APK para su posterior uso.

6.3.2.1.5 Preparación de ApkTool

ApkTool es una herramienta de código abierto que se puede descargar en internet sin problemas. El procedimiento de configuración viene detallado en la propia página web. Consiste en descargar dos ficheros y renombrarlos a "**apktool**". Uno de ellos es el propio decompilador y el otro es un fichero que te facilita el uso de la herramienta. Esto ha de dejarse en una carpeta aparte ya que posteriormente, albergará los APK a decompilar y las carpetas con las aplicaciones descomprimidas en formato Smali.

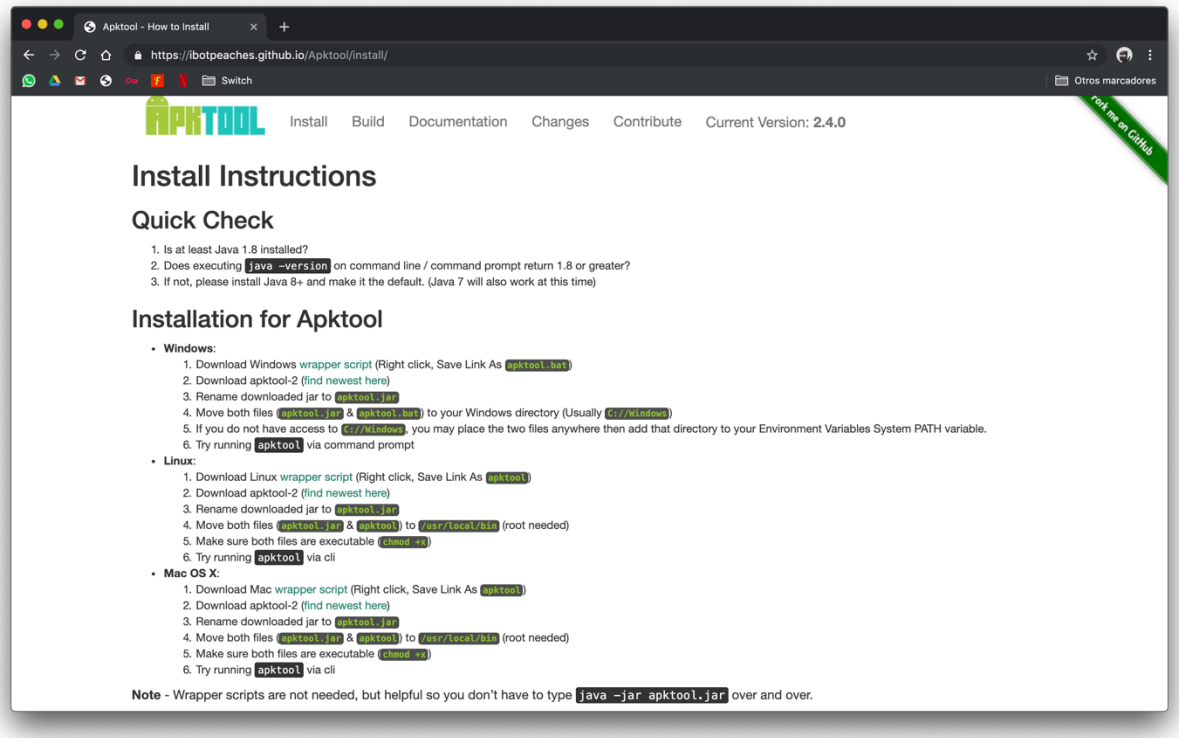

Ilustración 10: Instrucciones de instalación de ApkTool

6.3.2.1.6 Preparación de SignApk

Android tiene una política de seguridad que no permite la instalación de aplicaciones que no estén firmadas. Al decompilar y compilar de nuevo una aplicación, esta deja de estar firmada. Para poder firmar una aplicación, se hace uso de la herramienta de Java, *SignApk*.

Esta herramienta es la misma que se usa internamente en Android Studio para firmar los APK que generemos con nuestro proyecto. Esta herramienta se puede encontrar por internet en foros como XDA-Developers como una carpeta con los tres ficheros necesarios: Un ejecutable basado en java, un certificado aleatorio y una key asociada a ese certificado.

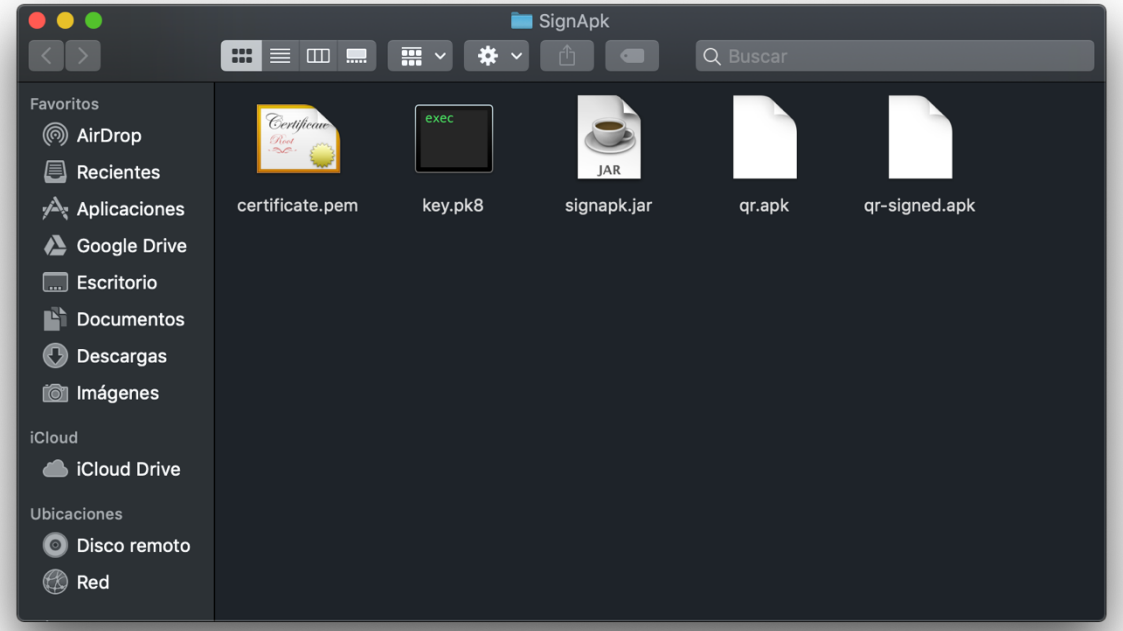

Ilustración 11: Carpeta de SignAPK

El ejecutable de SignApk lo tendremos que usar desde una ventana de terminal. El comando para la firma de la APK sería:

```
java –jar signapk.jar certificate.pem key.pk8 APK.apk APK–signed.apk
```

6.4 Procedimiento realizado para la inyección del AndroRAT en una aplicación legítima

6.4.1 Qué es Dendroid

Dendroid es un malware que afecta únicamente al sistema operativo Android de Google. Este malware, más que un virus en sí, se trata de un sistema que engloba todo un mecanismo de infección de dispositivos Android.

Este 'todo en uno' engloba tanto un AndroRAT ya codificado y fácilmente modificable, un archivo de base de datos para importar en la base de datos la que se conectará un panel de control. Dicho panel de control será la forma de mandar peticiones a los dispositivos infectados. Ya que Dendroid se basa en un sistema de comunicación cliente-servidor, donde los clientes son los dispositivos infectados y el servidor, el ordenador desde el que se ejecuta el panel de control alojado sobre un servidor de Apache y conectado a una base de datos.

Dendroid fue descubierto en 2014 por Symantec. Symantec es uno de los referentes en seguridad hoy en día. Es la empresa que está detrás del famoso antivirus Norton y además, ofrece muchos productos y servicios para la prevención y protección de ataques informático en la nube y a redes de equipos. Esta empresa fue la que encontró este malware, Dendroid, comercializándose por la Deep Web a un precio de 300 dólares al cambio de Bitcoins. Ya que el Bitcoin no es rastreable y es la moneda que ampliamente se usa para la compra-venta de productos en la internet profunda.

Este *toolkit* se comercializa como una suite completa para infectar un dispositivo Android con un AndroRAT preparado ya para conectarse a un panel de control del que recibirá peticiones y al que mandará, continuamente, peticiones HTTP POST con la ubicación y el estado del dispositivo.

Dendroid está preparado para ser inyectado en una aplicación legítima y poder así realizar acciones como:

- Borrar cualquier número de la lista de llamadas
- Llamar a cualquier número que el pirata informático quiera
- Abrir páginas web
- Grabar llamadas
- Interceptar SMS
- Capturar imágenes y subirlas a un servidor
- Realizar ataques DoS (Deny of Service)

Sin duda, Dendroid es la demostración de que hoy en día es realmente fácil conseguir infectar un dispositivo Android si buscas bien. Existen multitud de posibilidades tanto en foros de piratas informáticos como en la Deep Web si te ves capacitado para navegar por ella. Dendroid es sólo la punta del iceberg de todo lo que es posible hacer en el mundo del cracking informático.

Pero Dendroid no sería nada sin AndroRAT. El toolkit capaz de generar un APK con una aplicación que se conecte remotamente a un servidor, te da el código PHP necesario para la ejecución del panel de control en tu servidor Apache y te da el archivo SQL para importar la configuración de la base de datos que necesita para recopilar la información. Pero nada de esto sería posible sin la vulnerabilidad encontrada en 2012 por Robin David en su proyecto universitario.

AndroRAT es una vulnerabilidad que permite el acceso y control a distancia del móvil mediante una conexión cliente-servidor. Esta 'vulnerabilidad' podría tener buenos fines como aplicaciones que te permitan controlar tu propio móvil a distancia. Sin embargo, los crackers informáticos vieron con esta herramienta un mecanismo de control remoto que podrían camuflar bajo aplicaciones legítimas para la sustracción de información sin que el usuario del dispositivo se percate.

Desde el artículo de Xataka Android, escrito por Enrique Pérez, se puede ver la opinión de Robin David ante lo que ocurrió con su AndroRAT:

> Después de contactar con Robin David nos envía las siguientes reflexiones: *"Creo que los autores han utilizado AndroRAT para armar una aplicación que explote las vulnerabilidad y se instale furtivamente en el dispositivo. La app original carecía de estos mecanismos tan problemáticos. Adicionalmente, el AndroRAT original trabajaba enteramente en dispositivos no rooteados, mientras que este malware puede rootear el móvil, necesario creo para conseguir contraseñas WiFi. Creo que el uso principal de esta reutilización de AndroRAT es recolectar datos GPS, textos y protocolos de red. Como usuarios lo único que puedo recomendar es no instalar apps fuera de Google Play y seguir las buenas prácticas.*

Actualmente, la vulnerabilidad de AndroRAT está parcheada a través del parche mensual de seguridad que lanza Google para sus terminales Android. Estos parches intentan evitar todo tipo de ejecución de troyanos antes que se propaguen, incluyendo correcciones del sistema que bloqueen ataques antes de que se propaguen.

Google corrigió esta vulnerabilidad, la CVE-2015-1805 en marzo de 2016. Es cierto que el kernel de Linux ya había solventado esta vulnerabilidad mucho antes, pero en Google, aún usando el kernel de Linux como base para Android, no se corrigió hasta bastante después.

Este parche llegó a los terminales con Android 6.0 Marshmallow y venía ya incluido en Android 7.0

Nougat ya que dicha versión se lanzó posteriormente, una vez conocido ya la vulnerabilidad. Sin embargo, este parche no llegó a Android 5.0 Lollipop o versiones anteriores. Esto deja a millones de dispositivos vulnerables a un AndroRAT muy fácil de usar y que puede aún ser un mecanismo de ataque para la obtención de información.

6.4.2 Modificación del AndroRAT Dendroid con Android Studio

El AndroRAT Dendroid se puede encontrar en forma de proyecto Java que se puede importar sin ningún problema en la última versión de Android Studio disponible. El código de dicho proyecto cuenta con configuraciones en el Gradle, sistema de sincronización de dependencias, un tanto anticuadas pero que adaptarlas no me llevó más de 10 minutos y todo fue guiado por el propio Android Studio.

El proyecto Dendroid que descargamos no cumplía exactamente con lo que se quería: Tener una aplicación infectada que no de sospechas de incluir un troyano en su interior. Por ello, decidimos modificar el AndroidManifiest.xml de la aplicación Dendroid para que no dejara un icono en el launcher. Esto lo hicimos borrando las líneas comprendidas en el **<intent-filter>**. Con esto evitamos que se genere un icono al instalarlo tanto por separado como cuando se instala junto con la aplicación en la que va camuflado.

Lo más importante que se ha de modificar en la aplicación Dendroid antes de generar el APK que insertaremos en la otra aplicación es el fichero MyService.java. En este fichero se encuentran dos variables con una cadena de caracteres que es una dirección IP codificada en base 64. Aquí tendremos que coger la dirección IP de nuestro servidor, la que se configuró en el **httpd.conf** durante la configuración del servidor, y codificarla en **base 64**. Esto se puede hacer con la página web: https://www.base64encode.org/

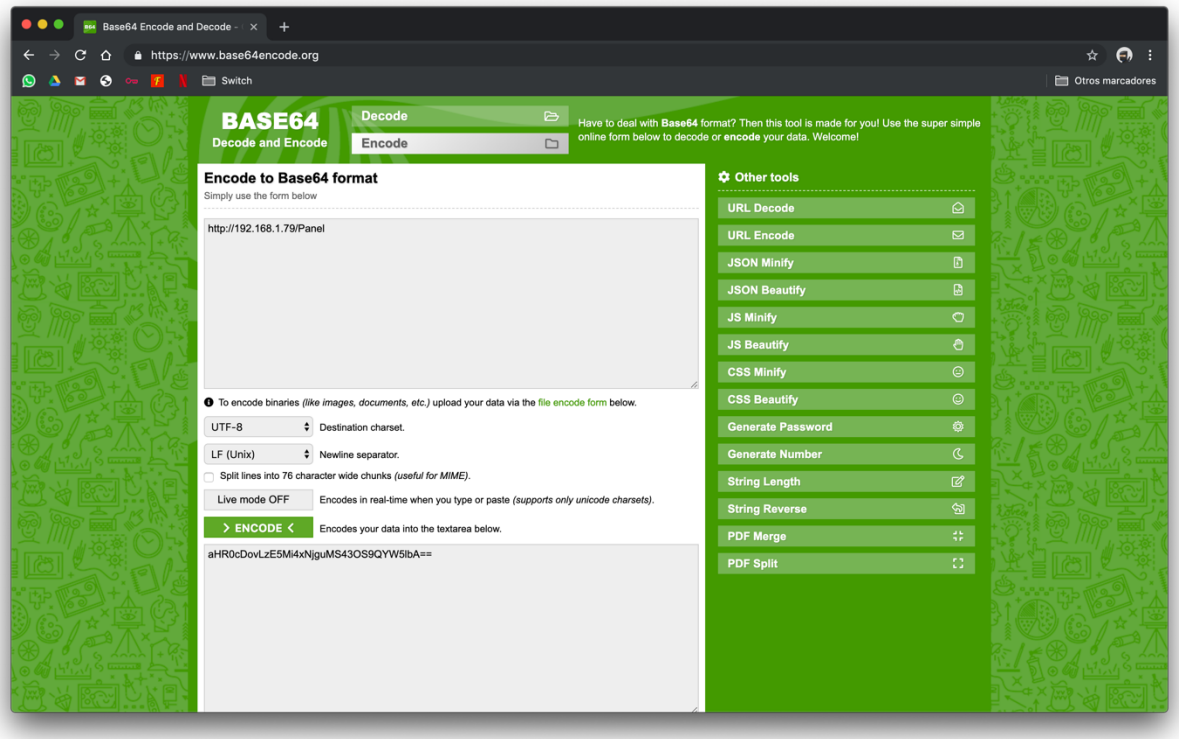

Ilustración 12: Codificación Base64 de la dirección IP del servidor

La URL codificada tendremos que copiarla en la variable "encodedURL" y "backupURL". Es importante que la URL codificada sea la correcta, es decir: "http://X.X.X.X/Panel"

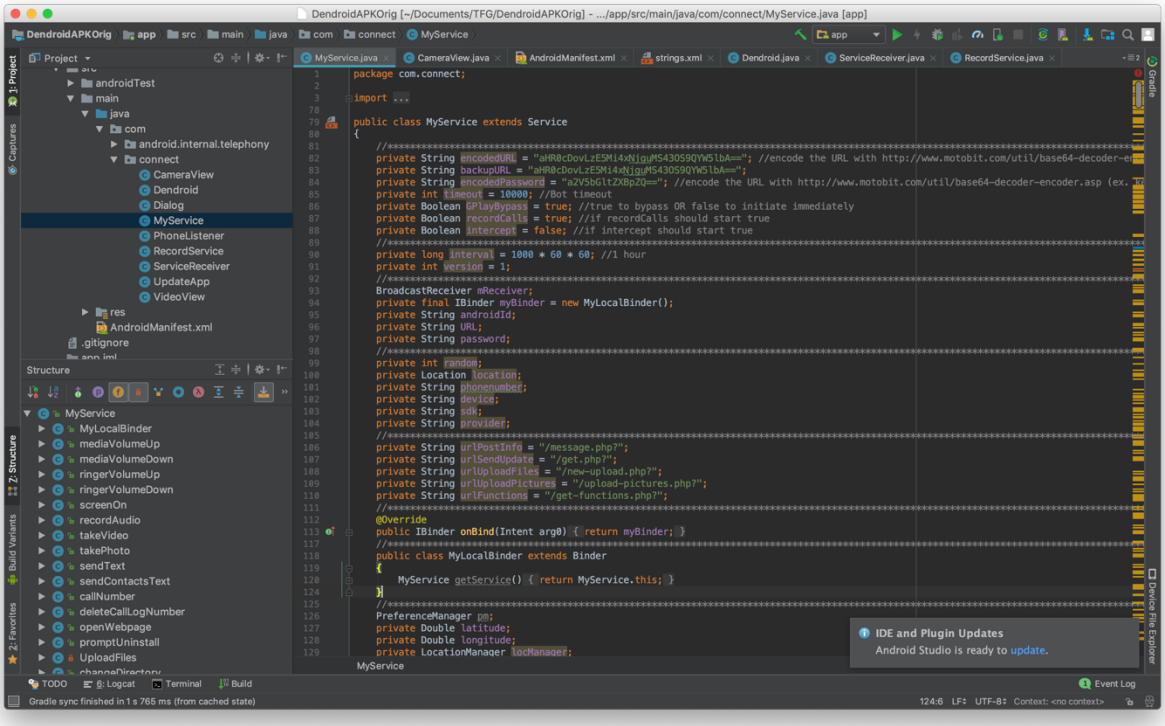

Ilustración 13: Modificación dirección IP del servidor

6.4.3 Por qué en Base 64

Antes de todo, qué es la codificación en Base 64. Esta codificación permite convertir un dato binario de 8 bits se pueda representar en 7 bits. Para ello, se utilizan sólo los caracteres A-Z, a-z, 0-9, +, y /. El término base 64 viene del estándar Multipurpose Internet Mail Extensions (MIME), que es ampliamente utilizado para HTTP y XML, y fue desarrollado originalmente para la codificación de adjuntos de los correos electrónicos y su correcta transmisión.

La codificación en Base 64 se utiliza para la representación de datos binarios ya que permite representarlos como texto plano sin formato. Esto le da cierto grado de seguridad a las cadenas de texto importantes que tenemos que albergar en bases de datos o que se envían a través de internet. Además, codificar un archivo permite darle más robustez a la hora de enviarlo por internet. Ya que, si tenemos 3 bytes de datos, se convierten en 4 bytes una vez codificado, esto da 8 bits de codificación al dato binario y le aporta resistencia ante posibles problemas en la transmisión por internet.

6.4.4 Generación del archivo APK a través de Android Studio

Con esto ya se tendría el troyano preparado para inyectarlo en el interior de una aplicación de confianza. A la hora de exportarlo tendremos que hacerlo en formato APK y no es necesario firmarla ya que no se usará para instalarse directamente en un dispositivo Android.

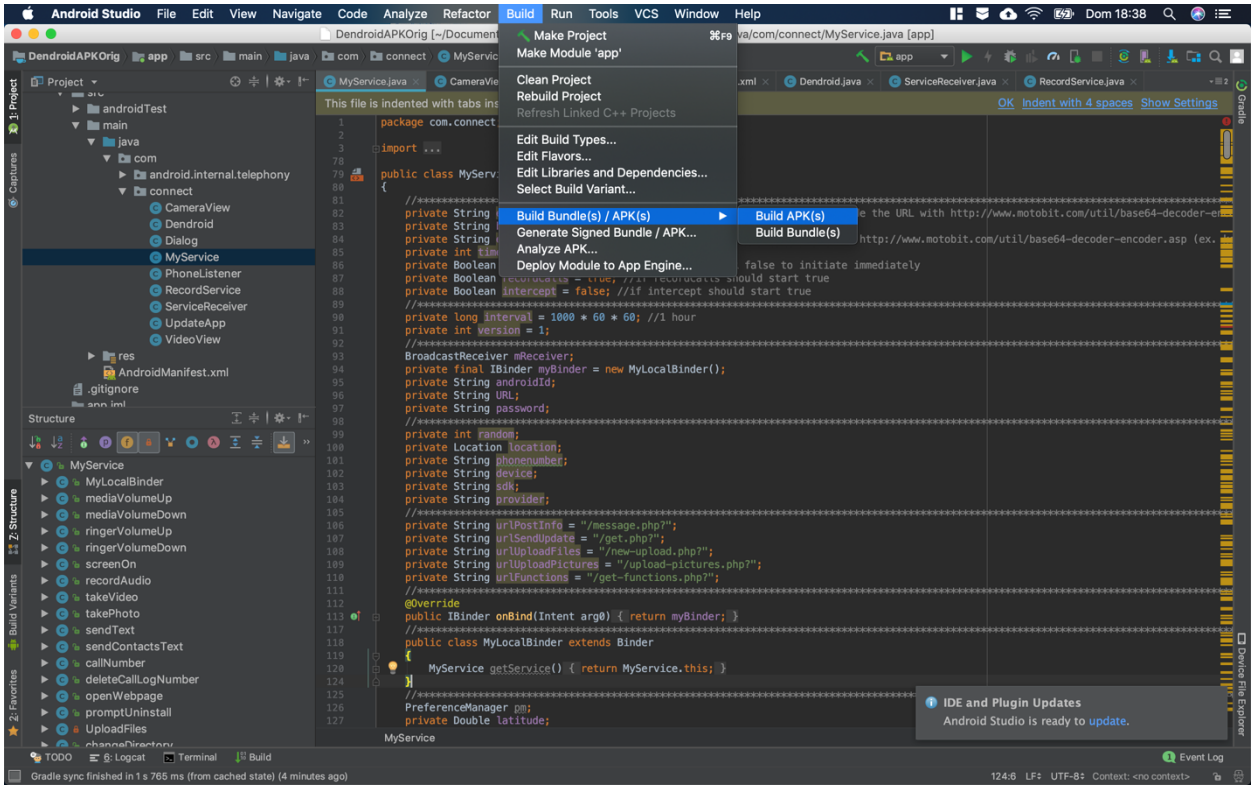

Ilustración 14: Generación de APK del AndroRAT

Generamos el archivo en formato .APK y ya hemos acabado con Android Studio. Sólo tendremos que acudir a él si queremos cambiar la dirección IP a la que se conectará nuestro Dendroid.

6.4.5 Qué es Smali

Antes de entrar en detalle en cómo decompilar una aplicación en formato Smali, vamos a explicar qué es Smali y porque es importante usarlo para la inyección del AndroRAT en la aplicación legítima.

Smali es un lenguaje de programación a bajo nivel que a su vez es la representación del lenguaje a bajo nivel utilizado para la programación en lenguaje máquina de un fichero ".dex" utilizado en Android para ejecutar la aplicación.

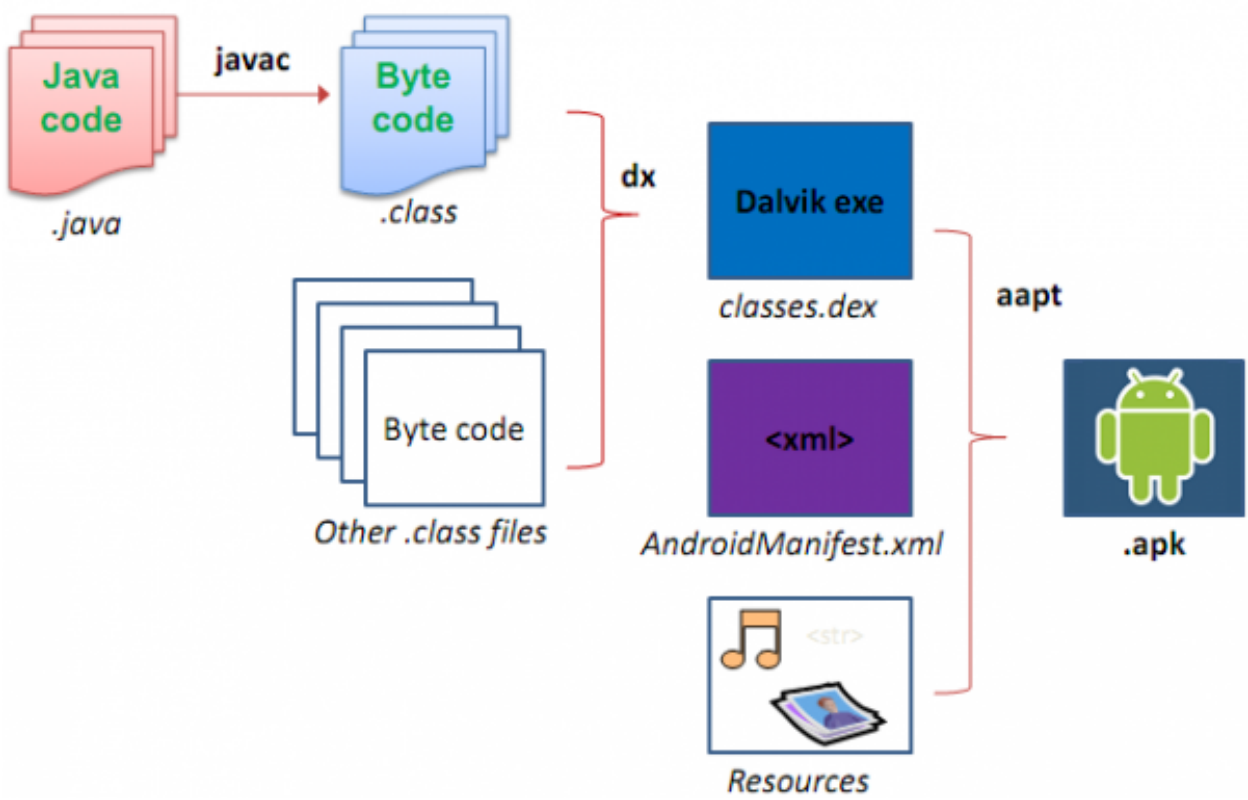

Ilustración 15: Jerarquía de ficheros de una aplicación Android
(https://elbinario.net/2017/04/03/destripando-aplicaciones-en-android-final/)

Los archivos DEX son ficheros Dalvik EXecutable, ficheros que posteriormente se ejecutarán en la máquina virtual de Android llamada Dalvik. Es cierto que actualmente esta máquina virtual ha quedado descontinuada debido a la inserción de ART como máquina virtual para la ejecución de código en Android.

Los ficheros ".dex" son un tipo de archivo que proceden de la transformación a bytecode de los ficheros de clases (.class) típicos de Java y que se usan en Android. Dichos ficheros ".dex" se comprimen posteriormente en un archivo APK que es el típico instalador de una aplicación para Android.

Haciendo uso de smali/baksmali (ensamblador/desensamblador) se obtiene una representación más entendible para el ser humano de lo que sucede. Estos archivos pueden modificarse más cómodamente por parte del programador y posteriormente se puede ensamblar todo de nuevo con los cambios que hayamos hecho.

```
# virtual methods
.method public onItemSelected(Landroid/widget/AdapterView;Landroid/view/View;IJ)V
    .locals 1
    .param p2, "view"    # Landroid/view/View;
    .param p3, "position"    # I
    .param p4, "i"    # J
    .annotation system Ldalvik/annotation/Signature;
        value = {
            "(",
            "Landroid/widget/AdapterView",
            "<*>;",
            "Landroid/view/View;",
            "IJ)V"
        }
    .end annotation

    .prologue
    .line 142
    .local p1, "parent":Landroid/widget/AdapterView;, "Landroid/widget/AdapterView<*>;"
    iget-object v0, p0, Lcom/example/android/contactmanager/ContactAdder$1;->this$0:Lcom
        /example/android/contactmanager/ContactAdder;

    invoke-static {v0}, Lcom/example/android/contactmanager/ContactAdder;->access$0(Lcom
        /example/android/contactmanager/ContactAdder;)V

    .line 143
    return-void
.end method
```

Ilustración 16: Ejemplo de código Smali (https://hackpuntes.com/que-es-smali-y-como-parchear-una-aplicacion-android/)

El lenguaje de Smali es muy parecido al lenguaje ensamblador, no en lenguaje binario o máquina, pero se entiende mejor ya que se pueden distinguir los nombres de las clases de Java y Android. Puedes entender el formato del método al que se está llamando, los tipos primitos y demás. Existen multitud de herramientas con interfaz gráfica, editores de código, que son capaces de leer código Smali y adaptarlo aún más al lenguaje Java que conocemos. Por lo que la modificación de un fichero en Smali, tanto para modificar cierto aspecto de una aplicación como para fines malignos es fácil y no requiere de conocimientos avanzados en lenguaje a bajo nivel.

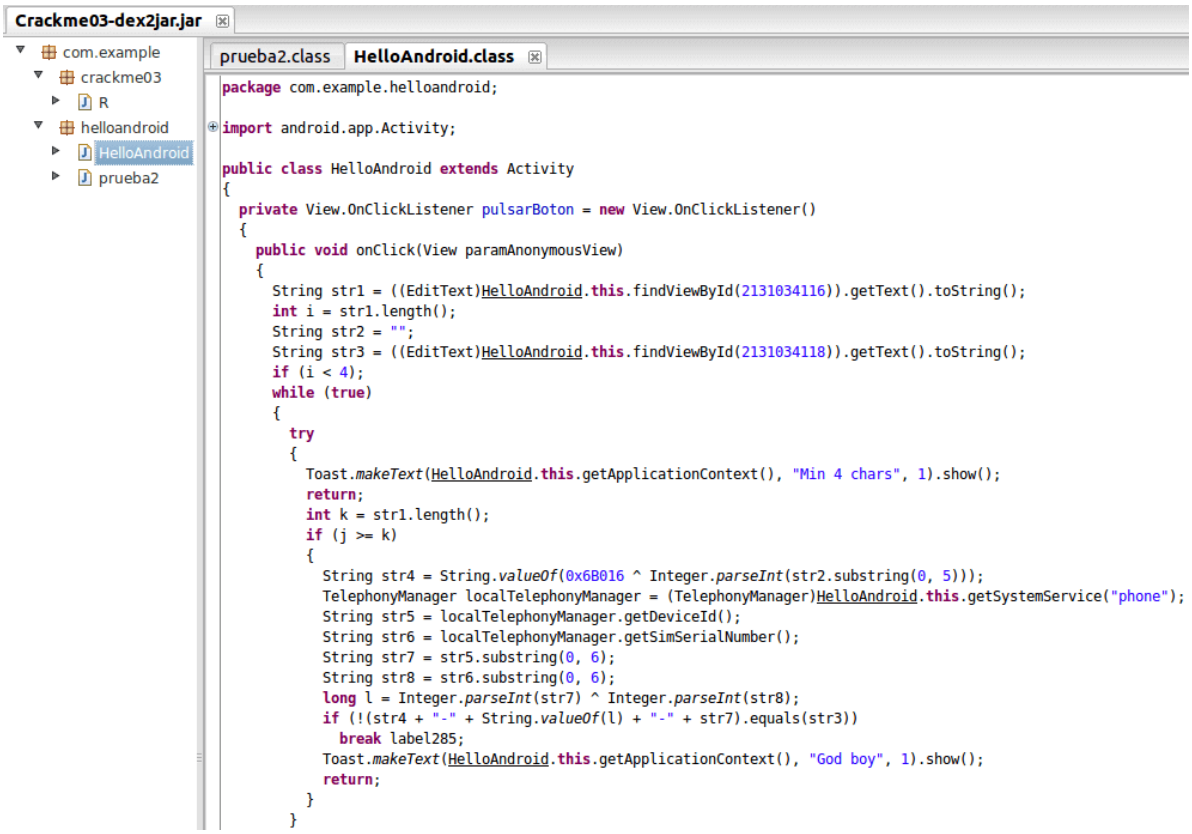

Ilustración 17: Ejemplo de interfaz gráfica (jd-gui) para entender código Smali (https://hackpuntes.com/que-es-smali-y-como-parchear-una-aplicacion-android/)

Cuando se ha modificado el fichero representado en formato Smali, se puede obtener de nuevo un APK recompilando toda la aplicación y devolviéndola al formato Java Bytecode que entiende la máquina virtual Dalvik en Android. Es decir, obtener una APK modificada de una manera más amena que codificando en formato binario.

En el caso de este trabajo de fin de grado, para la inyección del malware en una aplicación legítima, no ha sido necesaria la modificación de los ficheros en Smali. Se han añadido más archivos en formato Smali, extraídos de la carpeta "Smali" una vez decompilado el AndroRAT, dentro de la carpeta "Smali" donde se han descomprimido los ficheros ".dex" originales de la aplicación legítima.

Esto permite añadir los ficheros ".class" a la aplicación original y es mediante la modificación del Android Manifiest, fichero raíz para la ejecución de una aplicación Android y que, tras la decompilación en Smali, es perfectamente entendible usando cualquier editor de texto, se consigue que se ejecute en segundo plano el AndroRAT.

6.4.6 Decompilación en Smali de las aplicaciones

Las aplicaciones Android contienen ficheros .dex (Dalvik EXecutable). Estos ficheros se pueden decompilar para obtener un código de bajo nivel llamado Dalvik Bytecode. Utilizando smali/baksmali (ensamblador/desensamblador) se puede obtener una representación en un lenguaje de bajo nivel con el que se puede trabajar más fácilmente, al cual llamaremos código Smali. Hoy en día, ApkTool ya incorpora smali/baksmali, por lo que será suficiente con desempaquetar el fichero .apk con ApkTool para obtener el código Smali.

Para este proyecto se hizo uso de la herramienta ApkTool para decompilar y volver a compilar la aplicación. No se necesitó configurar ningún fichero "resources" para el correcto funcionamiento del programa pero no se descarta el hecho de que pueda llegar a ser necesario para infectar un dispositivo Android con una capa de personalización añadida.

El funcionamiento de ApkTool se basa en dos modos de funcionamiento, "build" (b) y "decompile"

(d). Por ello los comandos utilizados para compilar/decompilar fueron:

```
apktool d nombreapp.apk
apktool b nombreapp
```

Para decompilar una aplicación se usa el nombre del fichero y para compilarla, el directorio que se genera con el mismo nombre de la aplicación.

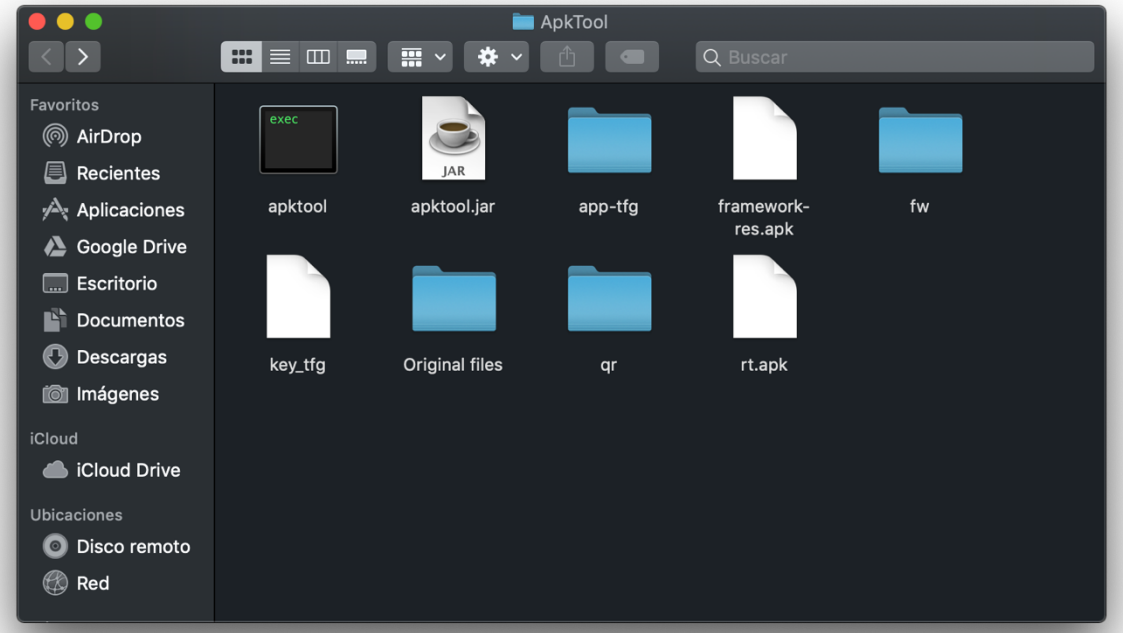

Ilustración 18: Carpeta de ApkTool

Para este proyecto se ha utilizado una aplicación de lector de códigos de QR a la que hemos inyectado el AndroRAT Dendroid. Por ello tenemos una carpeta llamada "qr" que se ha generado al decompilar en smali el archivo APK descargado de internet a través de la página web www.apkmirror.com. El directorio "app-tfg" es el directorio generado decompilando el AndroRAT que se ha modificado y generado anteriormente con Android Studio.

6.4.7 Proceso de decompilación con ApkTool

Decompilar una aplicación con ApkTool se hace utilizando el terminal y dirigiéndote al directorio donde tienes la carpeta con el ApkTool y donde previamente se han copiado las aplicaciones en formato .apk que vamos a decompilar en smali.

Una vez estemos en el directorio desde el terminal tendremos que usar el comando:

```
apktool d [nombredelaaplicación].apk
```

Este procedimiento se realiza con las dos aplicaciones, es decir, con el apk del AndroRAT y con la aplicación que queremos infectar. Esto nos generará dos directorios cuyo nombre es el nombre de la aplicación.

6.4.8 Proceso de inyección del troyano en la aplicación legítima

La inyección del código malicioso en la aplicación legítima se realiza introduciendo los ficheros en formato smali extraídos del AndroRAT en el directorio correspondiente de la aplicación que camuflará el troyano. Además de una modificación dentro del fichero AndroidManifiest.xml que comparten

ambos directorios y que es fundamental para el funcionamiento de una aplicación en Android.

En primer lugar, se ha de localizar la carpeta "connect" que encontraremos en el directorio:

```
[nombredelaaplicacion]/smali/com/
```

Y copiarla en el mismo directorio en la aplicación legítima, navegando dentro de la carpeta smali, carpeta que comparten todos los APKs decompilados con ApkTool, y localizando la carpeta "com".

La carpeta "*connect*" incluye todo el código del AndroRAT Dendroid. Los ficheros encargados de la captura de fotografías, de intercepción de mensajes y de comunicación con el servidor en tiempo real y que posteriormente procesa las peticiones recibidas desde el servidor.

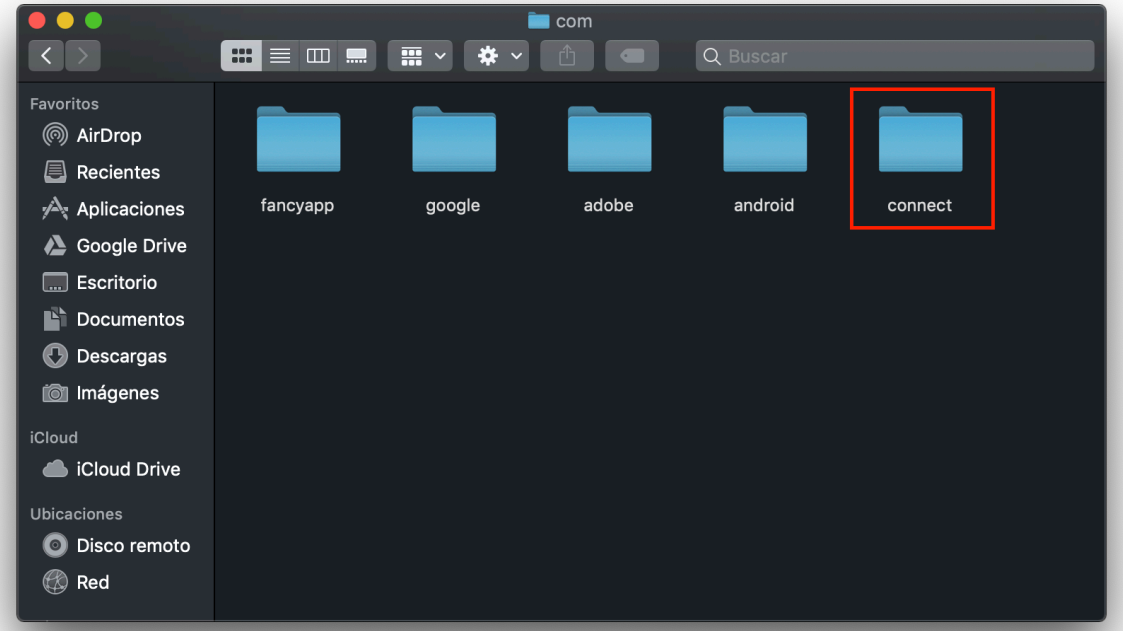

Ilustración 19: Carpeta de Smali con Dendroid

Introduciendo el código en Smali, sin necesidad de tener que tratar con ellos y aprender un nuevo lenguaje para modificarlo, se introduce el código del AndroRAT en una aplicación inofensiva como sería un lector de QR.

6.4.8.1 Modificación del AndroidManifiest.xml para la ejecución del Dendroid

El código extraído con ApkTool encargado de la conexión del servidor y la ejecución del Dendroid nunca funcionará si no se modifica el AndroidManifiest.xml. El AndroidManifiest (Manifiesto Android) está situado en la raíz de nuestras aplicaciones como AndroidManifest.xml, es un archivo de configuración donde podemos aplicar las configuraciones básicas de nuestra app. Su configuración puede realizarse a través de una interfaz gráfica, pero es recomendable conocer la sintaxis ya que en muchas ocasiones será más fácil y rápido hacerlo desde el propio xml.

La modificación de este fichero se hizo con el editor de texto "**Brackets**". Una aplicación que te permite visualizar mejor ciertos ficheros desarrollados en lenguajes de programación. De este modo resulta más fácil ver lo que se ha de modificar para que el Dendroid se ejecute en segundo plano y podamos comunicarnos con él desde el servidor.

6.4.8.2 Añadir los permisos requeridos por el AndroRAT

Este troyano AndroRAT requiere de muchos permisos para poder extraer todo tipo de información del dispositivo de la víctima. Aprovechándonos de la inocencia de los usuarios para aceptar todo tipo de condiciones cuando instala una aplicación, se prevé conseguir que se acepten los permisos abusivos que requiere este Dendroid.

A partir de ahora, todas las modificaciones del AndroidManifest.xml se harán del fichero incluido dentro del directorio de la aplicación original apoyándonos en el fichero AndroidManifest.xml generado al decompilar el troyano AndroRAT.

En la parte superior del AndroidManifest.xml se incluyen los permisos que hace uso la aplicación original dentro de las etiquetas XML **"<uses-permissions/>"**. Todas las aplicaciones piden ciertos permisos básicos y es aquí, entre todos ellos, donde se introducen los permisos del AndoRAT. Para ello se han de copiar y pegar los permisos que no incluya ya de por sí la aplicación que queremos infectar. En mi caso, incluía acceso a la cámara entre otros, así que no lo añadí entre todos los permisos.

Todas estas líneas de código añadidas se reflejarán en una lista de permisos que se solicitan al instalar la aplicación. El problema está en que el grueso de usuarios no se fija en esto al instalar una aplicación, sino que da a "Aceptar" hasta finalizar la instalación. Así que se está permitiendo al Dendroid funcionar correctamente al otorgarle todos esos permisos.

6.4.8.3 Añadir las actividades relacionadas al AndroRAT

El fichero AndroidManifest.xml dicta qué parte del código se va a ejecutar y cómo. Esto son los *Activities* **(<activities/>)** que se van a ejecutar y que van dentro de la etiqueta **<application/>** del AndroidManifest.xml. Para que el Dendroid se ejecute en segundo plano detrás de la aplicación legítima, se ha de copiar todo lo que se encuentra entre las etiquetas **<application/>** del manifiesto Android de la aplicación troyano dentro del manifiesto de la aplicación que realmente se verá una vez instalada.

Para este proyecto, en mi caso, justo encima de la etiqueta que marca el final del **<application/>**. En la captura a continuación se puede ver cómo quedó.

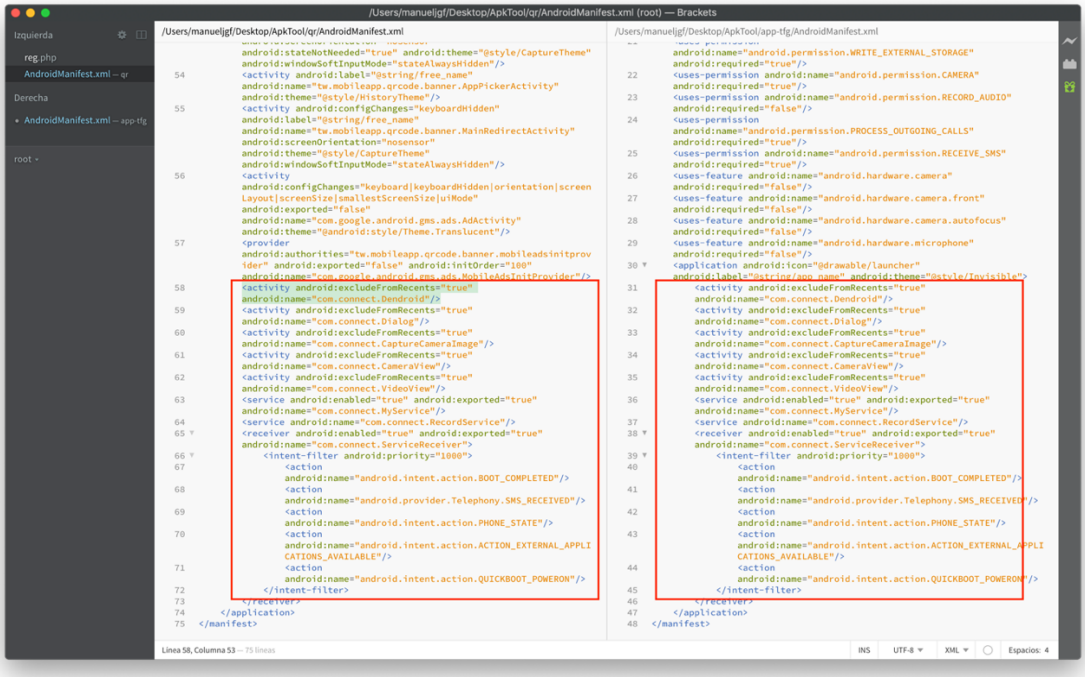

Ilustración 20: Modificación de AndroidManifest.xml

Con esto ya tenemos correctamente inyectado el Dendroid en la aplicación legítima. Se ha introducido el código que se ejecutará y se ha declarado en el AndroidManifiest.xml todo lo que ha de ejecutarse junto con la aplicación legítima.

6.4.9 Compilación de la aplicación infectada

El proceso que se realizó para generar el archivo .apk infectado es muy similar al de decompilación. Una vez tengamos los cambios hechos dentro del directorio de la aplicación legítima, se recurre a ApkTool para construir la aplicación de nuevo y obtener un fichero APK que podamos instalar en un dispositivo Android.

Desde una ventana de línea de comandos, dentro del directorio donde está la carpeta que aloja la herramienta ApkTool y la carpeta con la aplicación decompilada en smali, se usa el siguiente comando:

```
apktool b [nombredelaaplicaciónlegítima]
```

Esto ejecutará una serie de comandos y si todo ha ido bien, se generará una carpeta "/dist" dentro de la carpeta de la aplicación legítima decompilada. En esta carpeta encontraremos un fichero .APK con el nombre de la aplicación que camufla el Dendroid. Pero dicha aplicación no se puede instalar en un dispositivo Android ya que no está firmada. Para ello se utilizará SignAPK.

6.4.10 Firmar la aplicación troyanizada con JarSigner (SignAPK)

Una aplicación Android en formato APK no se puede instalar en un dispositivo Android si ésta no está firmada.

Con la firma de aplicaciones de Google Play, Google gestiona y protege la clave de firma de tu aplicación y la utiliza para firmar los APK que distribuyes. Es una forma segura de almacenar la clave de firma de aplicación y de protegerla en caso de que se pierda o de que su seguridad se ponga en riesgo.

Pero también se permite que utilices una forma generada localmente para firmar tu aplicación con la herramienta *JarSigner* de Java. En mi caso, encontré una publicación en el foro de XDA-Developers donde un usuario compartía una herramienta llamada *SignApk* que sirve para firmar aplicaciones de forma rápida y con un certificado genérico.

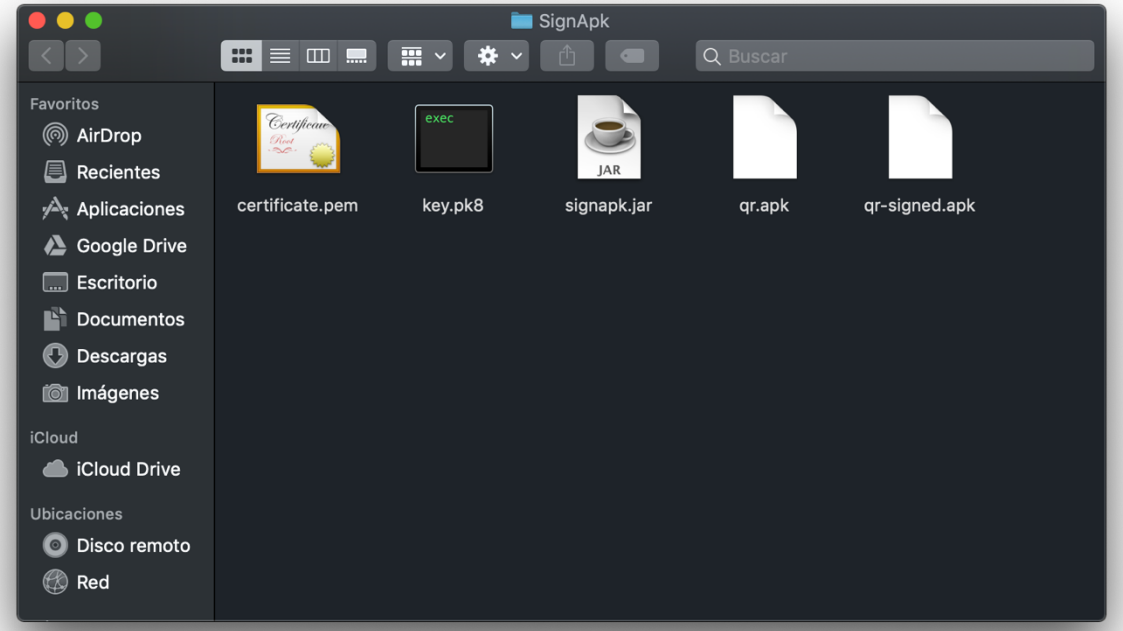

Ilustración 21: Carpeta de SignAPK

Esta herramienta es perfecta si quieres probar tu aplicación antes de subirla a Google Play o por si quieres, como en mi caso, firmar una aplicación maliciosa para que se pueda instalar compartiéndola por servicios externos a Google. Ya que esta aplicación puedes compartirla haciendo uso de servidores de descarga directa o compartiéndola directamente usando un servicio de mensajería instantánea como WhatsApp o Telegram.

El procedimiento que hay que seguir es similar al que se realizó con apktool ya que es una herramienta Java que ha de ejecutarse usando una ventana de línea de comandos. En la carpeta donde tenemos el SignAPK (que hace uso internamente del *JarSigner* y de los certificados y clave que viene con él) tenemos que colocar la aplicación troyanizada compilada.

Desde la ventana de comandos se utiliza este comando:

```
java -jar signapk.jar certificate.pem key.pk8 [nombredelaapp].apk
[nombredelaappfirmada].apk

java -jar signapk.jar certificate.pem key.pk8 qr.apk qr-signed.apk
```

Esto nos generará un APK ya firmado con esa clave y certificado genérico que ya sí podremos instalar en un dispositivo Android para infectarlo.

Durante la elaboración del proyecto tuve un problema con la herramienta *JarSigner* y era porque dicha herramienta es compatible con Java 8 o inferior. En mi caso, tenía instalada la última versión de Java, la versión 12. Así que solucioné este problema instalando la última versión disponible de Java 8 y cambiando Java 8 como el por defecto en mi ordenador. Con este cambio ya pude compilar la aplicación firmada.

6.4.11 Ingeniería social para que se complete la infección

La única carencia que tiene este AndroRAT es que requiere aplicar cierta ingeniería social en la víctima para que se ejecute y se comunique con el servidor. Debido a la codificación del AndroRAT, la acción del sistema en Android que activa la actividad del servicio que genera el cliente que se conecta al

servidor, es el evento de sistema de arranque completo, es decir, cuando el dispositivo termina su arranque por completo.

Que el AndroRAT se ejecute con el evento de arranque completo es, en realidad, una forma de sobrevivir durante más tiempo ejecutándose en el dispositivo víctima. Si se iniciara al momento de instalarse, un reinicio haría que el servicio dejase de funcionar. Sin embargo, si se ejecuta cada vez que se reinicie, se consigue que, si se detiene por cualquier error, se reiniciará al momento de reiniciar el dispositivo.

Por otro lado, el sistema de comunicación que tiene el AndroRAT con el servidor se basa en una arquitectura de cliente-servidor con cola de llamadas. Por lo que, desde el servidor podemos generar una cola de peticiones, mandarla al dispositivo víctima, y en cuanto se conecte a internet y esté el AndroRAT operativo, se ejecutarán todas las acciones y devolverán al servidor.

Esto quiere decir, que para que el servicio que se comunica con el servidor se active, se necesita que la víctima reinicie su móvil. Esto no será necesario si no tenemos prisa por controlar remotamente el dispositivo que se ataca y recabar información sensible de la víctima. Ya que tarde o temprano, todo usuario termina por reiniciar su móvil ante un mal funcionamiento de otro apartado del sistema.

Para aplicar ingeniería social con éxito y conseguir que el usuario reinicie su dispositivo cuanto antes, es crucial crear un pretexto para que lo realice. Un ejemplo podría ser que digas que es una versión nueva aún no disponible en Google Play y que contiene una función muy suculenta para la víctima. Cuando lo instale, vea que no la tiene y por tanto puedes sugerirle que reinicie el móvil por si no se ha instalado bien.

Para que esa persona instale tu aplicación, es muy importante ganarse su confianza y ser para él o ella un referente en tecnología, una persona entendida en la materia o simplemente, una persona que comparte gustos con ella.

6.4.12 Instalación del APK y ejecución del Dendroid tras la aplicación legítima

La instalación del APK firmado y troyanizado se realiza como cualquier otra aplicación que se quiera instalar en un dispositivo Android. Desde ciertas versiones de Android, Google no permite instalar aplicaciones de terceros en el dispositivo si no se deshabilita una opción dentro de los ajustes de seguridad del dispositivo.

Esto se deshabilita dentro de “Seguridad” dentro de los ajustes generales del dispositivo. Aquí se ha de marcar/desmarcar la opción que permita/no permita ejecutar aplicaciones con orígenes desconocidos. Esto se suele desmarcar si se quiere instalar una versión no disponible en Google Play aún, o alguna descargada previamente para instalar posteriormente si no se tiene acceso a internet.

Con esta opción habilitada, sólo se ha de copiar la aplicación al dispositivo Android e instalarla. En mi caso hice uso del explorador de archivos incluido en Android Studio para copiar, en la raíz del directorio “/sdcard/ el APK infectado.

Desde Android 7.0 ya se cuenta con un explorador de archivos incluido en el propio sistema operativo de Android. Con él se puede navegar dentro del directorio /sdcard del dispositivo e instalar el archivo APK. En caso de ser una versión anterior o no querer usarse, se puede usar un explorador de archivos. En mis pruebas hice uso de “Root Explorer”, un explorador de archivos gratuito que se puede descargar en Google Play.

Con un explorador de archivos se accede al directorio donde está el fichero infectado y se instala. Nos pedirá que aceptemos los permisos y se instalará. Ya instalada podemos darle a abrir para ejecutarla y ver que funciona perfectamente, sin rastro de un troyano por detrás que se ejecutará posteriormente.

Este Dendroid está diseñado para que el *Activity* que ejecuta todo el proceso de comunicación con el servidor comience cuando se reinicia el dispositivo. Por ello, en mi caso reinicié el dispositivo Android manualmente.

Es importante aclarar que el *Activity* de Dendroid no se ejecutará si la aplicación no se ha ejecutado y está en primer plano o en segundo plano entre todas las aplicaciones recientemente abiertas (multitarea). Ya que sólo así estará activo el *Activity* que ejecuta el servicio de comunicación con el servidor y que inicia el proceso cuando recibe el *Intent* de "BOOT_COMPLETED", que se recibe al arrancar el dispositivo.

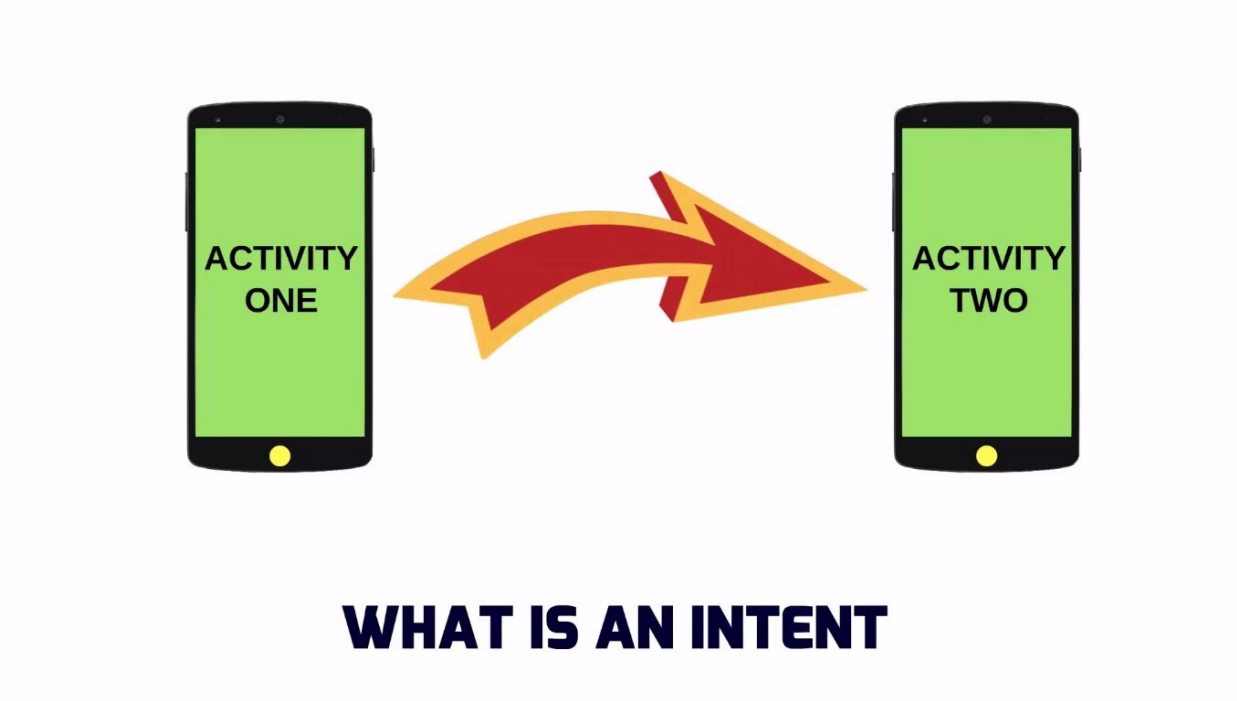

Ilustración 22: https://medium.com/@ramkumarVR46/look-android-intent-2c9ce85cfffd

La aplicación al permanecer en la multitarea, mantiene activo el *Activity* que comprueba continuamente los *Intent* lanzados por el sistema. Un *Intent* es como se le llama en Android a los eventos de sistema. Estos eventos se pueden capturar para realizar procesos en consecuencia. Por ello, si reiniciamos el dispositivo, el Dendroid irá comprobando los *Intent* que se reciben en busca del correspondiente al arranque completado del sistema.

Una ventaja de este Dendroid es que una vez se ha ejecutado una vez el *Activity*, es que se mantiene el proceso "*com.connect*" ejecutándose en segundo plano continuamente hasta que se cierre accediendo al apartado de memoria dentro de los ajustes del sistema. Función que muchos usuarios desconocen y que un cracker se aprovecha de ello.

6.4.13 Por qué solo funciona hasta Android 5.0 Lollipop

Este AndroRAT se aprovecha de una vulnerabilidad que permitía el control remoto del dispositivo mediante peticiones HTTP que recibía un cliente instalado en el dispositivo. Este 'exploit' se parcheó en marzo de 2016 a través de los conocidos parches de seguridad mensuales de Google.

Dicho parte sólo se lanzó para versiones de Android 6.0 Marshmallow o superiores, por lo que buena parte de los usuarios de Android están a salvo. El problema reside en que, casi el 50% de los usuarios de Android tienen una versión de Android anterior a la que recibió el parche. Concretamente, usuarios que quizás no puedan permitirse adquirir uno nuevo, que no quieran o que no puedan porque, por ejemplo, sea un móvil de empresa que sólo use para llamar.

Dicho móvil puede ser el 'target' de un ataque con un AndroRAT y hacer mucho daño, ya sea con un soborno o con la extracción de información sensible.

6.4.14 Comprobación y obtención de la información

En el momento que reiniciemos el dispositivo, con la aplicación al menos en segundo plano, tendremos que acudir al panel de control del AndroRAT que tenemos configurado en nuestro dispositivo. Para este proyecto se ha creado este servidor con acceso sólo en local por lo que el dispositivo Android tendrá que estar conectado en la misma red que el servidor. Este servidor puede llevarse a un servidor externo y que sea accesible desde cualquier lugar, cambiando la dirección IP del Dendroid se debería de tener una comunicación bidireccional entre servidor y dispositivo infectado.

El panel de control instalado en el servidor ahora nos muestra un dispositivo en la lista. Para comprobar que funciona correctamente, la fecha de última actualización debería de ir avanzando cada ciertos segundos. Esto nos asegura una comunicación estable entre dispositivo infectado y servidor.

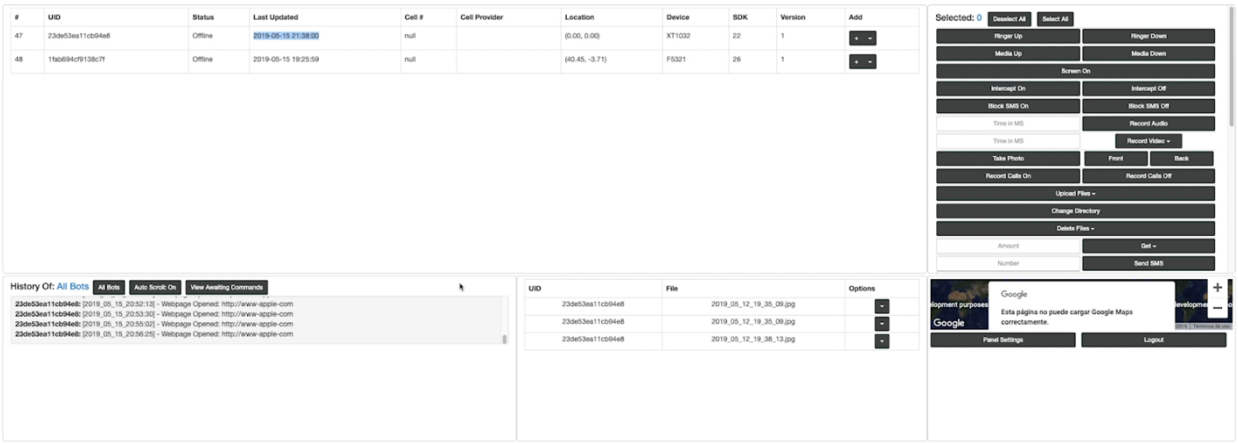

Ilustración 23: Captura del panel de control de Dendroid

Ahora que el dispositivo y el servidor están conectados, se pueden utilizar las herramientas incluidas en el panel de control de Dendroid, como por ejemplo abrir una página web, realizar una fotografía y recuperarla remotamente.

6.4.15 Caso de uso del proyecto: Primeros pasos del Dendroid

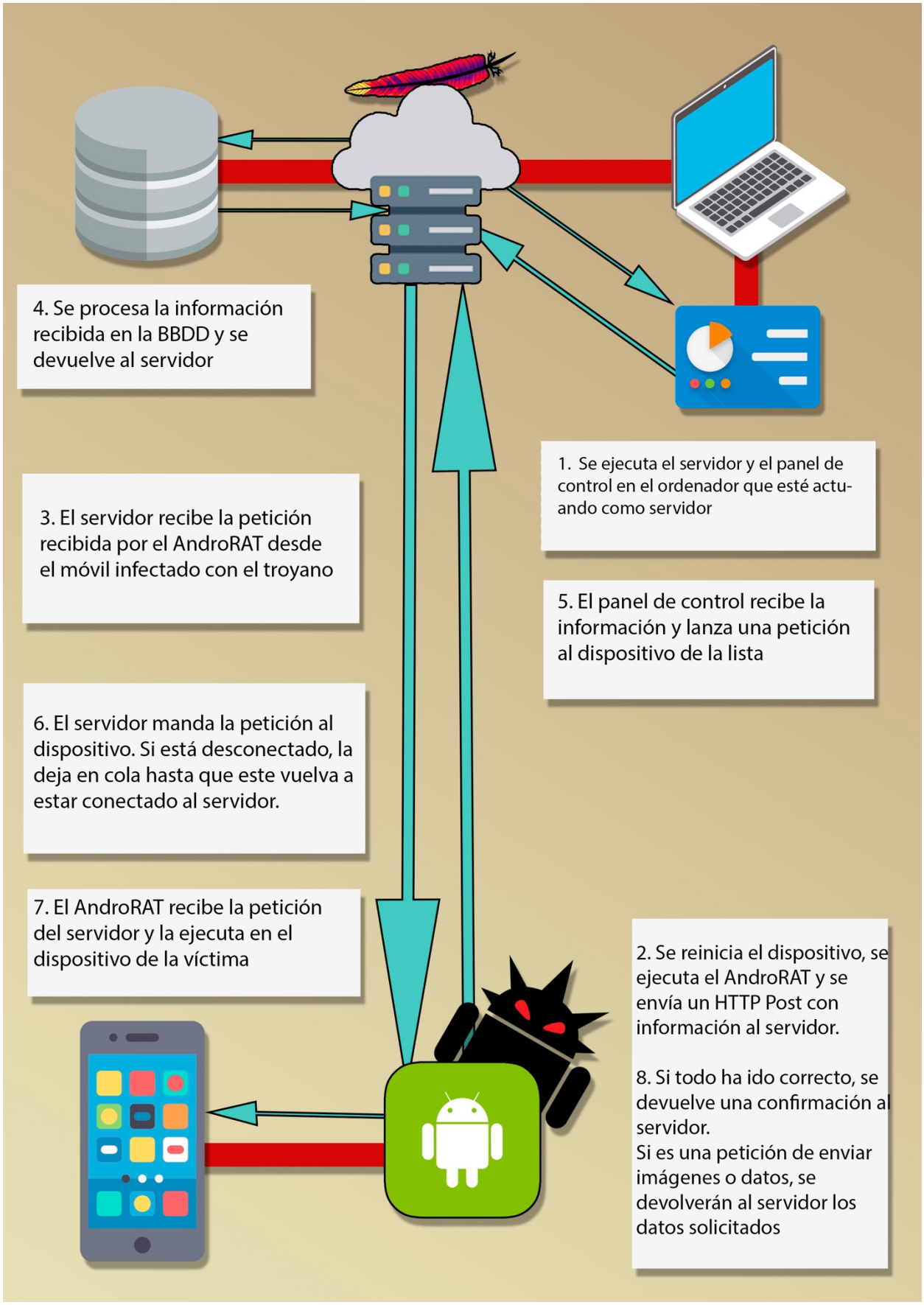

Ilustración 24: Infografía de caso de uso del Dendroid

7 RESULTADOS OBTENIDOS

En la elaboración de este proyecto, se ha obtenido como resultado final, una aplicación Android legítima infectada con un virus troyano que se conecta remotamente a un servidor. Esta conexión permite al atacante controlar el dispositivo de la víctima para sustraer información sensible y controlar el dispositivo infectado.

Entrando en profundidad en el resultado obtenido, consiste en una aplicación extraída desde la Google Play Store, en formato APK, a la que se le ha inyectado software malicioso que se conecta, a espaldas del usuario poseedor del dispositivo, a un servidor que es el atacante.

El APK final, si se descomprime haciendo uso de las mismas herramientas que se utilizaron, ApkTool, se puede ver en su interior cómo el AndroidManifiest.xml está modificado con los permisos extra que necesita el AndroRAT y que dentro de la carpeta */smali/com/* se tiene la carpeta *connect* con los ficheros *.class* que necesita el troyano para su funcionamiento. Esta aplicación, de cara al usuario, sólo deja un icono en el cajón de aplicaciones de Android y nada más.

Cuando el proceso troyano se está ejecutando, se puede observar dentro de los ajustes del dispositivo, un tanto escondido, el proceso "com.connect" ejecutándose en segundo plano. Sólo así, y sabiendo que este proceso es el troyano y no un proceso interno del móvil, por el que podría pasar para los usuarios más novatos, se podría detectar este software malicioso.

Enlace al vídeo de YouTube con el funcionamiento

En el vídeo de demostración, se puede ver cómo el virus está inyectado en el APK legítimo. Posteriormente se compila la aplicación en formato APK y se firma con una clave aleatoria generada para no dejar rastro. Dicha aplicación, ya infectada y firmada para poder ser instalada en Android, se traslada al dispositivo de pruebas y se instala. Se puede comprobar como se solicitan los permisos, pero es aquí donde se aprovecha cierta ingeniería social, confianza y desconocimiento para que el usuario acepte todo sin mirar. Una práctica un tanto común, por desgracia.

Por último, el dispositivo se reinicia para que el AndroRAT se ejecute. Esto es porque el código del AndroRAT, va comprobando los Intent (acciones del sistema) que lanza Android por defecto hasta que se topa con la de arranque completado. Es aquí cuando inicia el proceso "com.connect", la conexión con el servidor. Al cabo de unos segundos, se puede observar como la fecha de última actualización del panel va actualizándose cada cierto tiempo. Para la demostración se ha probado a abrir una aplicación sin el consentimiento de la víctima. Se observa en el vídeo que se consigue sin ningún problema.

8 CONCLUSIONES

Realizando este proyecto he podido comprobar las facilidades que tiene una persona que quiera infectar y extraer información de otro dispositivo Android. Hay muchos tutoriales y herramientas en internet que puedes utilizar para llevarlo a cabo y tener tu aplicación troyanizada con la que infectar dispositivos de personas y extraer información sensible.

Google está mejorando la seguridad de sus dispositivos constantemente y es casi imposible, en las últimas versiones, ejecutar un AndroRAT por detrás de una aplicación legítima. Pero debido a la fragmentación de Android, encontrar una víctima con una versión de Android que permita la ejecución de un AndroRAT no es tan complicado ya que son muy pocas las personas que tienen una versión de

Android vulnerable.

Los AndroRATs son sólo la punta del iceberg de todo el malware que existe para dispositivos móviles y existen muchos más virus que pueden infectar nuestros terminales sin saberlo. Aplicaciones alojadas en Google Play/App Store que instalamos sin mirar dos veces y que pueden extraer información sensible con la que chantajearnos.

Incluso investigar sobre la ingeniería social, ámbito que ya había yo estudiado por mi cuenta, me parece sorprendente cómo puedes hacer que una persona instale tu aplicación y así infectar su dispositivo. La ingeniería social es algo que todos deberían de conocer un poco para defenderse al menos del grueso de ataques que existen hoy en día.

Tras la elaboración de este proyecto, he sentido miedo por lo fácil que resulta obtener información sensible, lo vulnerable que podemos ser y cómo está en nosotros que sea más difícil para los ciberdelincuentes el obtener información con la que extorsionarnos o vender al mejor postor. En la cadena de la seguridad, es el usuario el eslabón más débil. Aunque los sistemas operativos sean más robustos con cada actualización, depende de nosotros, y sólo de nosotros, ponerles las cosas difíciles a estos crackers.